computation

MDPI

Article

Algorithm for Propeller Optimization Based on Differential Evolution

Andry Sedelnikov *, Evgenii Kurkin, Jose Gabriel Quijada-Pioquinto, Oleg Lukyanov, Dmitrii Nazarov, Vladislava Chertykovtseva, Ekaterina Kurkina and Van Hung Hoang

Institute of Aerospace Engineering, Samara National Research University, 34 Moskovskoe Shosse, 443086 Samara, Russia; eugene.kurkin@mail.ru (E.K.); hosekihada@yandex.ru (J.G.Q.-P.); lukyanov.oe@ssau.ru (O.L.); dvn69@mail.ru (D.N.); vladislaava.s@yandex.ru (V.C.); ekaterina.kurkina@mail.ru (E.K.); hunghoang2508@gmail.com (V.H.H.)

* Correspondence: axe_backdraft@inbox.ru

Abstract: This paper describes the development of a methodology for air propeller optimization using Bezier curves to describe blade geometry. The proposed approach allows for more flexibility in setting the propeller shape, for example, using a variable airfoil over the blade span. The goal of optimization is to identify the appropriate geometry of a propeller that reduces the power required to achieve a given thrust. Because the proposed optimization problem is a constrained optimization process, the technique of generating a penalty function was used to convert the process into a nonconstrained optimization. For the optimization process, a variant of the differential evolution algorithm was used, which includes adaptive techniques of the evolutionary operators and a population size reduction method. The aerodynamic characteristics of the propellers were obtained using the similar to blade element momentum theory (BEMT) isolated section method (ISM) and the XFOIL program. Replacing the angle of geometric twist with the angle of attack of the airfoil section as a design variable made it possible to increase the robustness of the optimization algorithm and reduce the calculation time. The optimization technique was implemented in the OpenVINT code and has been used to design helicopter and tractor propellers for unmanned aerial vehicles. The development algorithm was validated experimentally and using CFD numerical method. The experimental tests confirm that the optimized propeller geometry is superior to commercial analogues available on the market.

Keywords: differential evolution; penalty function; Bezier curves; SHADE algorithm; CAPR method; lightweight pipelining; isolated sections method

Citation: Sedelnikov, A.; Kurkin, E.; Quijada-Pioquinto, J.G.; Lukyanov, O.; Nazarov, D.; Chertykovtseva, V.; Kurkina, E.; Hoang, V.H. Algorithm for Propeller Optimization Based on Differential Evolution. *Computation* **2024**, *12*, 52. https://doi.org/10.3390/computation12030052

Academic Editor: Demos T. Tsahalis

Received: 9 November 2023
Revised: 16 February 2024
Accepted: 19 February 2024
Published: 6 March 2024

1. Introduction

The propeller is one of the most important elements of an aircraft that provides lift and propulsion in the air. The relevance of propellers as a propulsion system has increased with the current growth in the production of unmanned aerial vehicles for various purposes around the world. The air propeller's design is an important part of aircraft design, beginning from the conceptual design stage [1]. An optimized propeller can significantly reduce emissions, improve acoustic response, and enhance performance [2]. The design of air propellers to minimize propulsion system energy costs began with Zhukovsky, Betz, and Goldstein [3,4] and was then refined by other scientists [5–8].

Propeller optimization is a complex, often multi-objective problem that requires consideration of many factors and constraints, including aerodynamic issues, strength, and acoustics [7,9–12]. The most common goals of propeller optimization from an aerodynamic perspective are the requirements of maximum thrust, maximum efficiency, and a balanced propeller based on Paretto optimality [13–15].

In recent years, computational fluid dynamics (CFD) using computer-aided engineering (CAE) systems has become an important method for propeller design and computation [16–18]. Therefore, the use of the finite volume method is irrational because it requires

large computational power. Some of the most commonly used propeller aerodynamic characteristics calculation methods currently are Blade Element Method (BEM), Blade Element Momentum Theory (BEMT), Vortex Lattice Method (VLM), and Computational Fluid Dynamics (CFD) [1,19]. For tip-loss correction, different methods are used [20], including the Goldstein function [21] and Glauert correction [22–24]. The BEM and BEMT methods, due to the fast calculation, have been the most used in optimization processes. The Isolated Section Method (ISM) is similar to BEMT method, based on the Zhukovsky propeller vortex theory, using the Goldstein function for tip-loss correction [25,26]. It should be noted that the accuracy of the BEM, BEMT and ISM methods depends on the precision of the propeller's cross-sections aerodynamic coefficients prediction. XFOIL code [27] based on the discrete vortex method is a common technique for predicting the airfoils aerodynamic characteristics in propeller design [24,28].

The methodology of propeller geometry parametrization is part of its the aerodynamic optimization problem. Bezier surfaces and curves have the possibility of generating complex shapes using some control points [29]. This feature makes the use of Bezier curves and surfaces suitable for describing propeller cross-sections (airfoils [30–32]) and the propeller blades in general, including chord and twist [1,33]. Due to the reduced number of control points, these curves and surfaces are used in propeller shape optimization processes [34], especially when used in conjunction with evolutionary algorithms [35,36] and artificial neural networks [1,33].

Various methods are used in solving the propeller optimization problem. The authors of [37] solved a multi-objective optimization problem subject to a set of constraints using a direct search algorithm. The paper [28] describes the experimental verification and application of a method of multi-criteria optimization using genetic algorithms for the design of a propeller for a high-altitude aircraft. The articles [38,39] discuss the effectiveness of differential evolution-based algorithms for optimizing the shape of different types of propellers.

BEMT and IMS methods for calculating a given shape propeller aerodynamic performance require an airfoil angle of attack selection cycle, causing the need for an airfoil drag polar calculation database [40] and drag curve extrapolation when the geometric twist exceeds the limits on angles of attack. Replacing the twist angle in the design variables with the angle of attack allows the optimization process to immediately search for the desired airflow field and eliminate an additional cycle of searching for the angle of attack for each propeller individual. In the current work, a modification of ISM consists in the airfoil effective angle usage instead of the geometrical twist angle is presented. The geometric twist is calculated for the optimum propeller when the optimum distribution of the angle of attack over the span is found. This approach also improves the robustness of the optimization process by directly controlling the constraints imposed on the angle of attack.

The parametric geometric model of the propeller used in the current work implements Bezier curves to specify the dependencies of the parameters of thickness, curvature, and positions of the maximum thickness and maximum curvature of the airfoil along the blade span. This makes it possible to use different thicknesses and curvatures of airfoils along the blade span, which allows the aerodynamic twist of the blade to be varied.

In addition, the article considers the use of interpolation of the calculation of the aerodynamic characteristics of the airfoils on the blade span, which also allows to accelerate the process.

The techniques of parameterization of propeller geometry, calculation of its aerodynamic characteristics and optimization of its shape considered in the paper are implemented in the OpenVINT code [41]. This code uses the following techniques: geometry modeling with Bézier curves, a modified ISM for obtaining propeller aerodynamic coefficients based on XFOIL for airfoils characteristics calculation, optimization by an evolutionary algorithm with adaptive techniques, and population size reduction methods.

The structure of this paper is as follows: Section 2 describes the methods that comprise the OpenVINT algorithm; in Section 3, two case studies are described in which the

versatility of the algorithm for optimizing different types of propellers is shown; finally, the discussions relevant to the results and conclusions of this work are presented.

2. Methods

2.1. Mathematical Model of Propeller Optimization

The objective of this paper is to develop an algorithm that allows the optimization of propeller geometry to minimize the power required to achieve a given thrust. This translates into improved energy consumption. Mathematically, the optimization problem can be expressed as a constrained optimization problem. This paper describes the development of an optimization algorithm to improve the performance of propellers used in unmanned aerial vehicles.

$$\begin{aligned} \min \quad & W(x), \\ \text{such that} \quad & T_{obj} - T(x) \leq 0, \\ & x \in X \end{aligned}$$

where W(x) is the required power of the propeller; T(x) is the thrust provided by the propeller; T_{obj} is the desired thrust; x is the vector of design parameters belonging to the feasible set of solutions X.

For this optimization problem to be solved by evolutionary algorithms, the problem had to be converted into an unconstrained optimization problem. This was achieved by using a penalty function, which will be used as a fitness function. The penalty function is expressed as follows:

$$L(x) = \begin{cases} W(x) & \text{if } \psi(x) = 0 \\ R\psi(x) + U^* & \text{if } \psi(x) > 0 \wedge W(x) \leq U^* \\ R\psi(x) + W(x) & \text{if } \psi(x) > 0 \wedge W(x) > U^* \end{cases} \quad (1)$$

where

$$\psi(x) = \max\left\{0, T_{obj} - T(x)\right\} \quad (2)$$

where U* is an upper bound on the constrained global minimum value; R is a penalty parameter that allows to equate the values obtained in ψ(x) with U*. A large value is assigned to U* initially, but the value needs to be updated with the current best known function value at feasible points. Hence, the initial U* is not updated until a feasible solution has been found. ψ(x) > 0 only if x is infeasible [42]. Therefore, the optimization problem remains as:

$$\begin{aligned} \min \quad & L(x), \\ \text{such that} \quad & x \in X \end{aligned}$$

2.2. Selection of Design Variables

Two types of design variables were proposed: those that describe in a general way the geometry of the propeller or the operation, such as the diameter of the propeller (d, in m), number of blades (B), and number of revolutions per minute (n_m, rev/min); and others that change depending on the radius of the propeller, such as the chord length, effective angle of the airfoil, and the geometry of the airfoil. We considered that these variables have a smooth and continuous variation along the propeller radius using Bezier curves. This variation was achieved using two quadratic Bezier curves [43] (see Figure 1). A quadratic Bezier curve is the path drawn by the following function:

$$BC(t) = (1 - t^2)P_0 + 2(1 - t)tP_1 + t^2P_2 \quad (3)$$

where P_0, P_1 and P_2 are control points, and t is a parameter that always varies from 0 to 1.

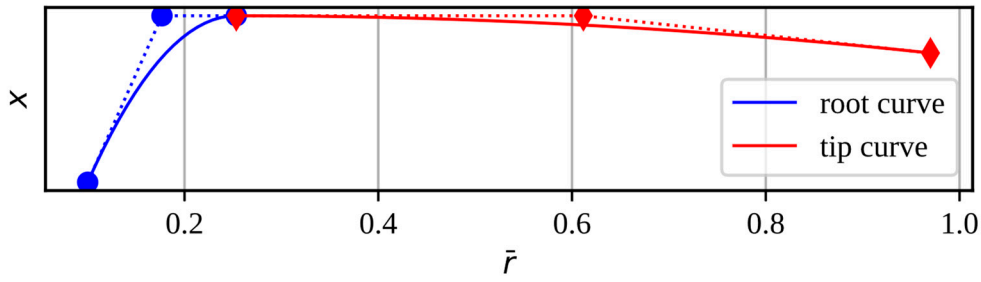


Figure 1. The Bezier curves determine the variation of the design variables.

BC(t) can also be decomposed as ($B_{\bar{r}}(t)$, $B_x(t)$), where $\bar{r}$ indicates the relative radius of the propeller and x the design parameter.

$$B_{\bar{r}}(t) = (1-t)^2\bar{r}_0 + 2(1-t)t\bar{r}_1 + t^2\bar{r}_2, \tag{4}$$

$$B_x(t) = (1-t)^2 x_0 + 2(1-t)tx_1 + t^2 x_2, \tag{5}$$

The control points that determine the Bézier curves for each design variable as a function of the propeller's relative radius are as follows:

- Root curve

$$\begin{cases} \bar{r}_0^r = 0.1 \\ \bar{r}_1^r = 0.5\bar{r}_{xm} + 0.05 \\ \bar{r}_2^r = \bar{r}_{xm} \end{cases} \quad \begin{cases} x_0^r = x_r \\ x_1^r = x_m \\ x_2^r = x_m \end{cases} \tag{6}$$

- Tip curve

$$\begin{cases} \bar{r}_0^t = \bar{r}_{xm} \\ \bar{r}_1^t = 0.5\bar{r}_{xm} + 0.485 \\ \bar{r}_2^t = 0.97 \end{cases} \quad \begin{cases} x_0^t = x_m \\ x_1^t = x_m \\ x_2^t = x_t \end{cases} \tag{7}$$

In Equations (4)–(7), x may be the chord relative to the propeller diameter (c/d), the effective angle (α), maximum thickness (y_t), position of the maximum thickness relative to the chord (x_t), the maximum camber (y_c), and the position of the maximum camber relative to the chord (x_c). Each of these variables is with respect to the profile of each section of the propeller blade. The variable $\bar{r}_{xm}$ refers to the union position of the two Bezier curves. The subscripts r, m, t refers to the values of the design variable with respect to 10% of the blade radius, the union point of the Bezier curves, and 97% of the blade radius, respectively. Therefore, each vector of design variables is formed as x_i = [$(c/d)_r$, $(c/d)_m$, $(c/d)_t$, $\bar{r}_{cm}$, α_r, α_m, α_t, $\bar{r}_{\alpha m}$, y_{tr}, y_{tm}, y_{tt}, $\bar{r}_{ytm}$, x_{tr}, x_{tm}, x_{tt}, $\bar{r}_{xtm}$, y_{cr}, y_{cm}, y_{ct}, $\bar{r}_{ycm}$, x_{cr}, x_{cm}, x_{ct}, $\bar{r}_{xcm}$, n_m, B, $d]_i$.

2.3. Development of the Input Geometry

The Isolated Sections Method (ISM) [44,45] was selected to obtain the aerodynamic characteristics of the different propeller configurations. This method requires knowledge of the geometry, chord, and effective angle of attack of the airfoil in different sections of the propeller blade. The chord values and the effective angle of attack were obtained by constructing the distribution curves (Bezier curve) and using the input parameters $(c/d)_r$, $(c/d)_m$, $(c/d)_t$, $\bar{r}_{cm}$, α_r, α_m, α_t, and $\bar{r}_{\alpha m}$. To obtain the coordinates of the airfoil in each section, an airfoil parameterization method based on the Bezier-PARSEC technique was proposed [46]. The use of Bezier curves makes it possible to model a wide variety of profile types [39,47]. Each airfoil is constructed with four cubic Bezier curves, two curves for determining the thickness of the airfoil and two curves for determining the camber of the airfoil (see Figure 2).

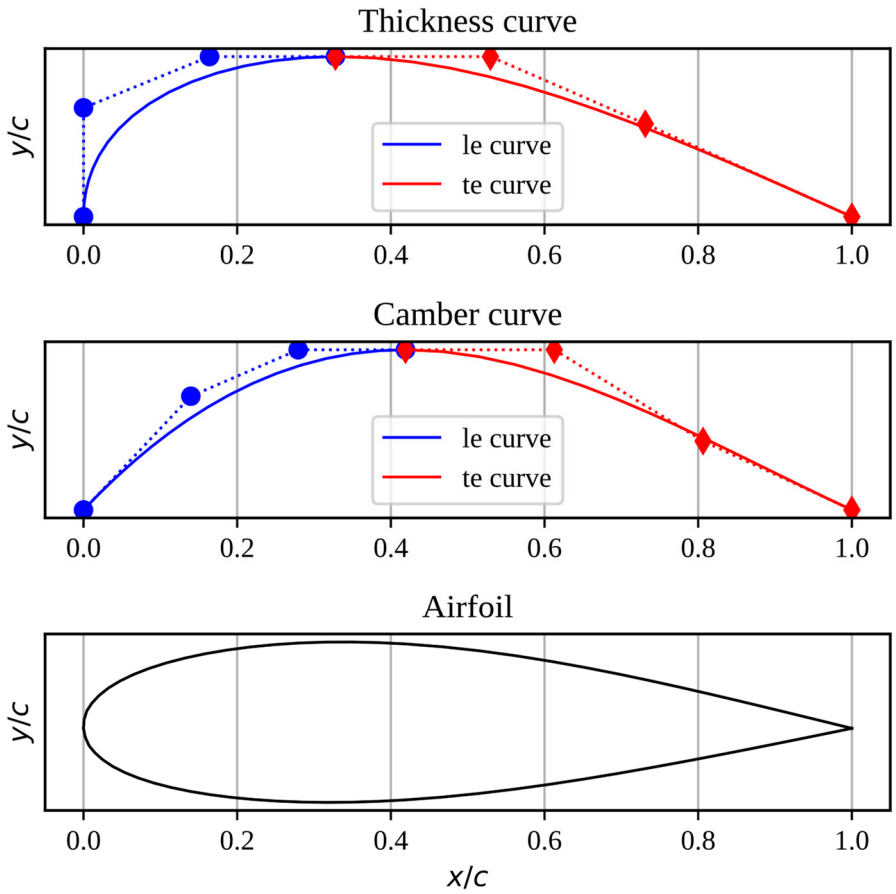


Figure 2. Airfoil construction using cubic Bezier curves.

The parametric functions for determining each cubic Bezier curve are as follows:

$$B_x(t) = (1-t)^3 x_0 + 3(1-t)^2 t x_1 + 3(1-t)t^2 x_2 + t^3 x_3, \tag{8}$$

$$B_y(t) = (1-t)^3 y_0 + 3(1-t)^2 t y_1 + 3(1-t)t^2 y_2 + t^3 y_3, \tag{9}$$

The construction of the profile is carried out using the following equations:

$$X_t = B^{le}_{xt}(t) + B^{te}_{xt}(t) \tag{10}$$

$$Y_t = B^{le}_{yt}(t) + B^{te}_{yt}(t) \tag{11}$$

$$X_c = B^{le}_{xc}(t) + B^{te}_{xc}(t) \tag{12}$$

$$Y_c = B^{le}_{yc}(t) + B^{te}_{yc}(t) \tag{13}$$

$$\theta = \tan^{-1}\left(\frac{dY_c}{dX_c}\right) \tag{14}$$

The control points for the leading edge thickness curve are defined by:

$$\begin{cases} x^{le}_0 = 0 \\ x^{le}_1 = 0 \\ x^{le}_2 = 0.5x_t \\ x^{le}_3 = x_t \end{cases} \qquad \begin{cases} y^{le}_0 = 0 \\ y^{le}_1 = 0.34y_t \\ y^{le}_2 = 0.5y_t \\ y^{le}_3 = 0.5y_t \end{cases} \tag{15}$$

The control points for the trailing edge thickness curve are defined by:

$$\begin{cases} x_0^{te} = x_t \\ x_1^{te} = 0.3 + 0.7x_t \\ x_2^{te} = 0.6 + 0.4x_t \\ x_3^{te} = 1 \end{cases} \quad \begin{cases} y_0^{te} = 0.5y_t \\ y_1^{te} = 0.5y_t \\ y_2^{te} = 0.29y_t \\ y_3^{te} = 0 \end{cases} \tag{16}$$

The control points for the leading edge camber curve are defined by:

$$\begin{cases} x_0^{le} = 0 \\ x_1^{le} = x_c/3 \\ x_2^{le} = 2x_c/3 \\ x_3^{le} = x_c \end{cases} \quad \begin{cases} y_0^{le} = 0 \\ y_1^{le} = 0.71y_c \\ y_2^{le} = y_c \\ y_3^{le} = y_c \end{cases} \tag{17}$$

And the control points for the trailing edge camber curve are defined by:

$$\begin{cases} x_0^{te} = x_c \\ x_1^{te} = (1 + 2x_c)/3 \\ x_2^{te} = (2 + x_c)/3 \\ x_3^{te} = 1 \end{cases} \quad \begin{cases} y_0^{te} = y_c \\ y_1^{te} = y_c \\ y_2^{te} = 0.43y_c \\ y_3^{te} = 0 \end{cases} \tag{18}$$

The points for determining the upper curve of the airfoil are determined by:

$$X_U = X_c - Y_t \sin\theta \tag{19}$$

$$Y_U = Y_c + Y_t \cos\theta \tag{20}$$

And the points for determining the lower curve of the airfoil are determined by:

$$X_L = X_c + Y_t \sin\theta \tag{21}$$

$$Y_L = Y_c - Y_t \cos\theta \tag{22}$$

Algorithm 1 shows the procedure for obtaining the input conditions for the ISM.

Algorithm 1: Subroutine for creation of the Bezier curves and airfoils

Inputs: x_i, d, $\bar{r}$, NS
Outputs: $(c/d)_i$, α_i, x_{ti}, y_{ti}, x_{ci}, y_{ci}, X_U, Y_U, X_L, Y_L
Create the distribution curve for $(c/d)_i$ as a function of the $\bar{r}_i$ of the blade with (4), (5), (6), (7);
Create the distribution curve for α_i as a function of the $\bar{r}_i$ of the blade with (4), (5), (6), (7);
Create the distribution curve for x_{ti} as a function of the $\bar{r}_i$ of the blade with (4), (5), (6), (7);
Create the distribution curve for y_{ti} as a function of the $\bar{r}_i$ of the blade with (4), (5), (6), (7);
Create the distribution curve for x_{ci} as a function of the $\bar{r}_i$ of the blade with (4), (5), (6), (7);
Create the distribution curve for y_{ci} as a function of the $\bar{r}_i$ of the blade with (4), (5), (6), (7);
//Get the airfoil in each section of the blade
for s = 1 **to** NS **do**
    Get $x_t(\bar{r}_{s,i})$, $y_t(\bar{r}_{s,i})$, $x_c(\bar{r}_{s,i})$ and $y_c(\bar{r}_{s,i})$;
    Get $X_{ts,i}$ and $Y_{ts,i}$ with (8), (9), (10), (11), (15), (16);
    Get $X_{cs,i}$ and $Y_{cs,i}$ with (8), (9), (12), (13), (17), (18);
    Get $\theta_{s,i}$ with (14);
    Get the points of the sth-airfoil X_U, Y_U, X_L, Y_L with (19), (20), (21), (22);
return Outputs

2.4. Construction of the Output Geometry

The final geometry of the blade is obtained after executing the ISM algorithm because this process is where the blade twist is obtained. Figure 3 shows the process of drawing a propeller blade from the proposed Bezier curves and the geometric torsion distribution obtained from Algorithm 2. To create the base shape of the blade, the torsion blade axis is first defined, which crosses the chord of each station at the point where the maximum thickness of the aerodynamic profile (x_t) is located. From this axis, the strings are rotated

according to the curve φ. On the other hand, with the four Bezier curves x_t, y_t, x_c, and y_c, the aerodynamic profiles of each station are generated using the procedure shown in Figure 2. Finally, the profiles are located and scaled according to the corresponding chord length.

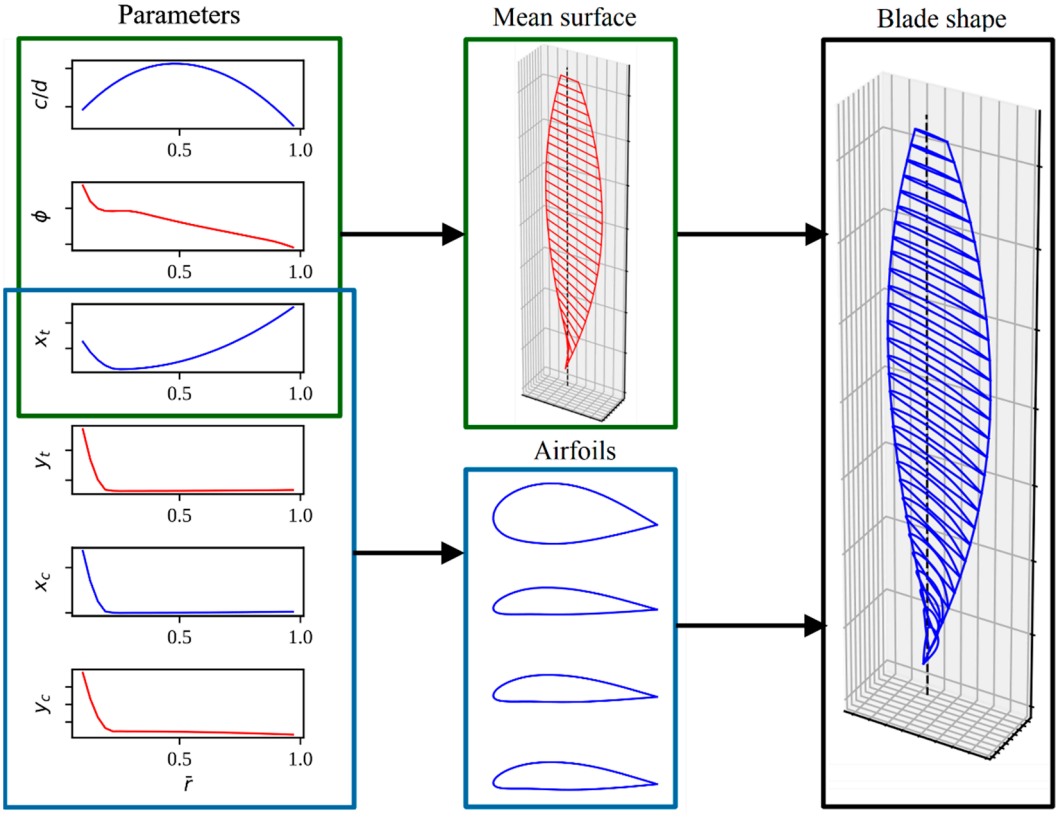


Figure 3. Drawing of a blade from Bezier curves and the geometric twist curve.

2.5. Isolate Section Method

The objective function of the optimization process is to calculate the propeller power (W, in W) required to achieve thrust (T, in N). A modified version of was used to obtain these values. The variation between the original and our modified version of the isolated section method (ISM) is the geometric twisting of the sections (φ, in deg) using the effective angle of attack of the sections (α, in deg) as the input value (Algorithm 2).

The input variables required by the method are as follows: B, V_∞, d, n (n, rev/s), and the physical characteristics of the air (density (ρ, in kg/m^3), kinematic viscosity (ν, [m^2/s]), and sound speed (a, [m/s])). The isolated section method requires sectioning one of the propeller blades into a finite number of sections (NS). In each section, it is necessary to know the following values: chord length (c, in m), $\bar{r}$, and the geometry of the airfoil ((X_U, Y_U), (X_L, Y_L), where each coordinate is in m).

The iterative method for obtaining the thrust provided by the propeller and the required power is described in [44]. This method was modified to use the effective angle of attack (α) of the profile in each section as a constant and the geometric twist (φ) of the section as a variable that is updated in each iteration.

Having the local characteristics of the flow, in addition to the chord, geometry, and angle of attack of the airfoil, coefficients of lift (c_l) and drag (c_d) coefficients in each section are calculated. The next step is to obtain $\overline{U}_1$, $\overline{V}_1$ and $\overline{\Gamma}_r$ by an iterative process, which is necessary to obtain the coefficients c_t and m_k. The equations involved in this process are as follows:

$$\overline{v}_1 = -\frac{\overline{v}}{2} + \sqrt{\frac{\overline{v}^2}{4} + \overline{u}_1(\overline{r} - \overline{u}_1) + 2\int_{\overline{r}}^{1} \frac{\overline{u}_1^2}{\overline{r}} d\overline{r}}, \tag{23}$$

$$\overline{U}_1 = \overline{r} - \overline{u}_1 \,, \tag{24}$$

$$\overline{V}_1 = \overline{v} + \overline{v}_1 \,, \tag{25}$$

$$\overline{W}_1 = \sqrt{\overline{V}_1^2 + \overline{U}_1^2} \,, \tag{26}$$

$$\beta_1 = \tan^{-1}\left(\frac{\overline{V}_1}{\overline{U}_1}\right), \tag{27}$$

$$\sigma = \frac{Bc}{R\pi} \,, \tag{28}$$

$$\overline{\Gamma}_r = \frac{1}{8}\sigma c_l \overline{W}_1 \,, \tag{29}$$

$$f_r = \frac{2}{\pi}\cos^{-1}\left(e^{-\frac{0.5B(1-\overline{r})}{\overline{r}\sin(\beta_1)}}\right). \tag{30}$$

Previously before initializing the iterative process, it is important to initialize the n values of $\overline{u}_1$ and $\int_{\overline{r}}^{1} \frac{\overline{u}_1^2}{\overline{r}} d\overline{r}$ equal to zero. At the end of each iteration $\overline{u}_1$ has to be updated making use by:

$$\overline{u}_1 = \frac{\overline{\Gamma}_r}{f_r \overline{r}} \,, \tag{31}$$

while the values of $\int_{\overline{r}}^{1} \frac{\overline{u}_1^2}{\overline{r}} d\overline{r}$ are updated as shown in lines 24, 25 and 26 of Algorithm 2, where trapz (Y, X) integrates along the given axis using the compound trapezoidal rule, Y is the input array to integrate and X is the sample points corresponding to the Y values.

It has been proven that the loop described between lines 10 and 26 of Algorithm 2 only needs 10 cycles to obtain good results.

To obtain the coefficients c_t and m_k, it is first necessary to calculate their differentials in each section, and then integrate with respect to $\overline{r}$, making use of trapz().

$$dc_t = 8\overline{\Gamma}_r\left(\overline{U}_1 - K^{-1}\overline{V}_1\right) \tag{32}$$

$$dm_k = 8\overline{\Gamma}_r\left(\overline{V}_1 + K^{-1}\overline{U}_1\right)\overline{r} \tag{33}$$

By obtaining the coefficients c_t and m_k, the thrust provided by the propeller (T) and the required power of the propeller (W) can be calculated.

$$T = 0.5 c_t \rho(\omega R)^2 \pi R^2 \tag{34}$$

$$W = 0.5 m_k \rho(\omega R)^3 \pi R^2 \tag{35}$$

Other propeller performance metrics that can be obtained with this method are the dynamic efficiency η_d and the static efficiency η_s of the propeller.

$$\alpha_p = \frac{T}{\rho n^2 d^4} \tag{36}$$

$$\beta_p = \frac{W}{\rho n^3 d^5} \tag{37}$$

$$\lambda_p = \frac{V_\infty}{nd} \tag{38}$$

$$\eta_d = \frac{\alpha_p \lambda_p}{\beta_p} \tag{39}$$

$$\eta_s = \frac{c_t^{3/2}}{2m_k} \tag{40}$$

Algorithm 2: Modified Isolate Section Method

Inputs: n, B, V_∞, d, ρ, ν, a, c, $\bar{r}$, $[X_U, Y_U, X_L, Y_L]$, NS
Outputs: T, W, η_d, η_s

R = d/2;
Get ω with (31);
Get $\bar{v}$ with (32);
for s = 1 **to** NS **do**
    $r_s = \bar{r}_s R$;
    Get Re_s and M_s in each section;
    c_{ls}, c_{ds} = runXFOIL(α_s, $[X_U, Y_U, X_L, Y_L]_s$, Re_s, M_s);
    $0 \rightarrow \bar{u}_{1_s}$, $0 \rightarrow \left(\int_{\bar{r}}^{1} \frac{\bar{u}_1^2}{\bar{r}} d\bar{r} \right)_s$
for t = 1 **to** 10 **do**
    $I_u = \emptyset$;
    for s = 1 **to** NS **do**
        Get $\bar{v}_{1_s}$ with (23);
        Get $\bar{U}_{1_s}$ with (24);
        Get $\bar{V}_{1_s}$ with (25);
        Get $\bar{W}_{1_s}$ with (26);
        Get β_{1s} with (27);
        $\varphi_s = \alpha_s + \beta_{1s}$;
        $K_s = c_{ls}/c_{ds}$;
        Get σ_s with (28);
        Get $\bar{\Gamma}_{r_s}$ with (29);
        Get f_{rs} with (30);
        Update $\bar{u}_{1_s}$ with (31);
        $\left(\frac{\bar{u}_{1_s}^2}{\bar{r}_s} \right)_s \rightarrow I_{u_s}$;
    for s = 1 **to** NS **do**
        Get $\left(\int_{\bar{r}}^{1} \frac{\bar{u}_1^2}{\bar{r}} d\bar{r} \right)_s$ with trapz(I_u[s : **end**], $\bar{r}$[s:**end**]);
for s = 1 **to** NS **do**
    Get dc_{t_s} with (32);
    Get dm_{k_s} with (33);
c_t = trapz(dc_t, $\bar{r}$);
m_k = trapz(dm_k, $\bar{r}$);
Get α_p, β_p and λ_p with (36), (37) and (38) respectively;
Get outputs T, W, η_d and η_s with (34), (35), (39) and (40) respectively;
Save φ;
return Outputs

2.6. Airfoil Aerodynamic Coefficients Calculation

The aerodynamic coefficients of each section of the blade was obtained by using the Xfoil program, which shows a good performance for the evaluation of profiles. In Figure 3, the experimental coefficients of the Clark-Y profile can be observed using XFOIL, OpenFOAM (with the k-ω SST turbulence model) and experimental tests.

In Algorithm 2, the modified ISM is described in pseudo code. To simplify the ISM calculations, it is necessary to obtain the radius (R, [m]), the angular velocity (ω, [rad/s]) and the relative velocity ($\bar{v}$, dimensionless) of the propeller.

$$\omega = 2\pi n \tag{41}$$

$$\bar{v} = \frac{V_\infty}{\omega R} \tag{42}$$

Sections are defined by the $\bar{r}$ array. For example, in the cases that we evaluate, the sections begin to be numbered from 10% of the R of the propeller and end at 97% of R ($\bar{r}$ = [0.1, . . ., 0.97]). In our modified ISM, having the α of each section as input, it is necessary to calculate the aerodynamic coefficients of each section delimited by $\bar{r}$. For this, it is necessary to obtain the local Reynolds (Re) and Mach (M) numbers.

To determine the aerodynamic characteristics of the propeller and reduce the number of calculations, interpolation of the profile data was performed over a smaller number of sections, in which the characteristics of the angle of attack were calculated considering the local Reynolds and Mach numbers (see Figure 4).

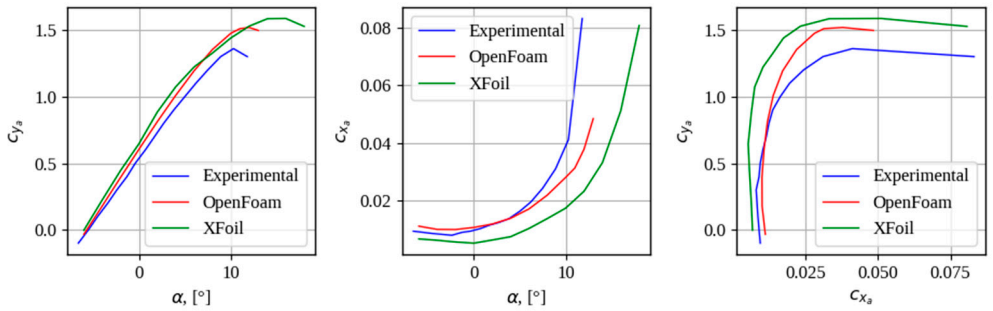


Figure 4. Aerodynamic coefficients of the CLARK Y profile by different methods.

A study was conducted out on the required number of sections, which showed that the use of 15 sections to calculate the aerodynamics of the profile and 75 sections to integrate thrust and moment allows one to quickly obtain accurate values of the propeller characteristics (Figure 5).

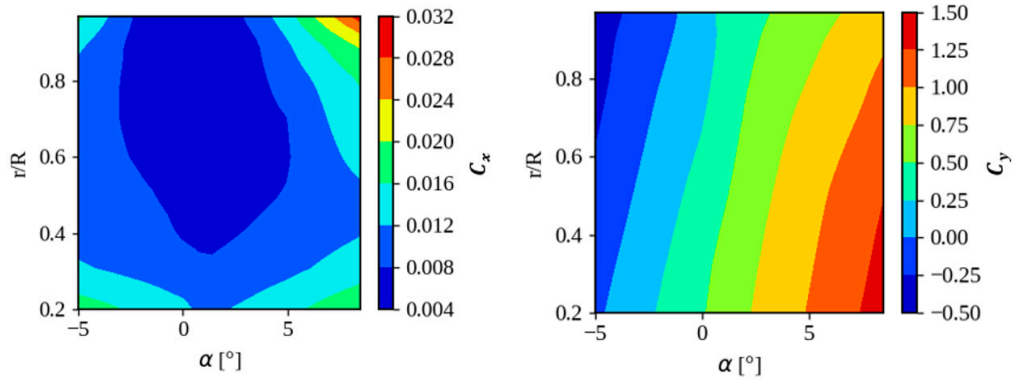
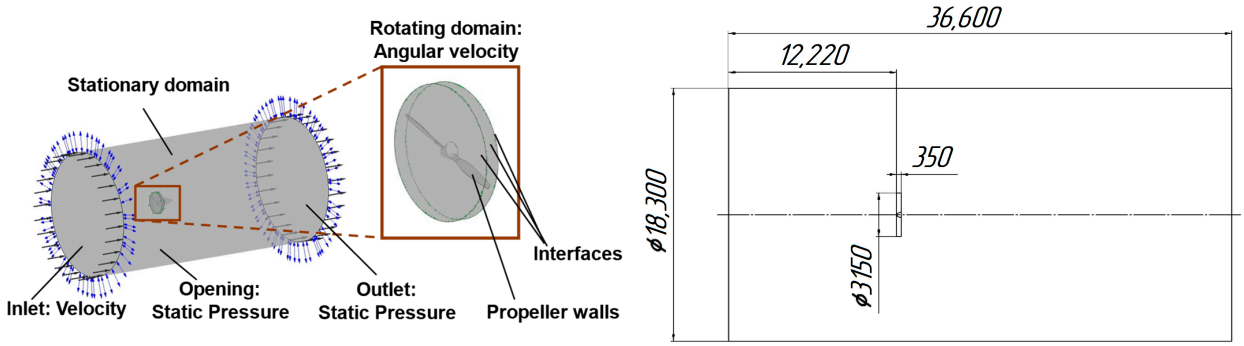


Figure 5. Calculation and interpolation of aerodynamic characteristics of the profile by propeller span.

2.7. Isolate Section Method Validation

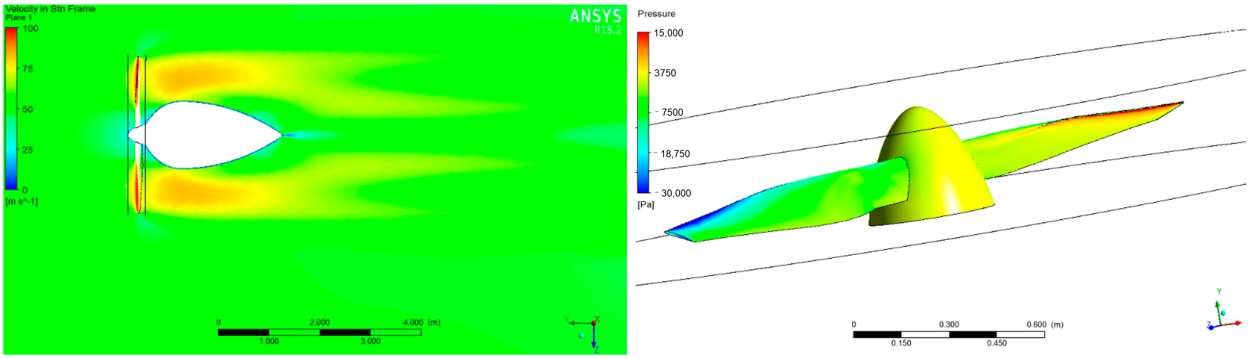
The proposed method of propeller calculation was carried out by comparing it with the experimental data of the NACA 5868–9 propeller (experimental results, geometric and kinematic characteristics of the propeller are given in [45]), as well as by comparison with the results of numerical mathematical modeling by solving the Navier–Stokes equations in ANSYS CFX 18.2 software using the shear stress transport turbulence model. The propeller diameter is 3.15 m. The problem statement and the dimensions of the static and rotational domains are shown in Figure 6. The configuration and dimensions of the domain are based on those shown in [48]. The stator mesh has 2,618,015 tetrahedral elements, and the rotor mesh has 11,130,389 elements (tetrahedra and pyramids). The minimum element sizes were 150 mm for the stationary domain, 15 mm for the rotating domain, 4.0 mm on the propeller wall. The first boundary layer thickness was 0.05 mm. In the CFX calculation, the propeller was modeled together with the nacelle, matching the geometry to the experimental conditions. The free wind speed is 170 km/h, with rotational speeds

from 800 rpm to 2500 rpm. Validation was performed for two cases of propeller geometry, with blade angles of $\varphi_{0.75} = 25°$ and $\varphi_{0.75} = 35°$. The values of the mean residuals for all equations do not exceed 10^{-3}. The velocity field and pressure distribution are shown in Figure 7. The comparison results in terms of thrust coefficient and power from the advanced ratio are shown in Figure 8.



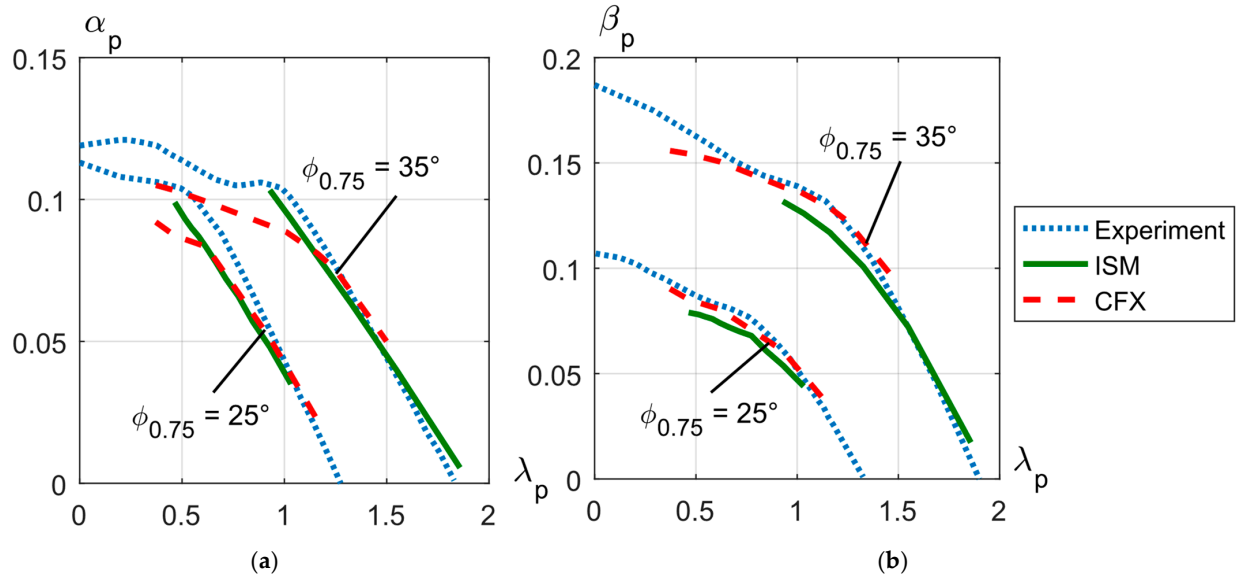
(**a**) (**b**)

Figure 6. NACA 5868-9 propeller calculation in ANSYS CFX, (**a**) the problem statement, (**b**) the domain dimensions.



(**a**) (**b**)

Figure 7. Example of NACA 5868-9 propeller calculation for 1500 rpm and $\varphi_{0.75} = 25°$ case, (**a**) velocity in stationary domain, (**b**) pressure at propeller wall.



(**a**) (**b**)

Figure 8. Comparison by thrust coefficient (**a**) and power coefficient (**b**).

2.8. Optimization Algorithm

The optimization algorithm used to solve the task is based on the Differential Evolution (DE) algorithm. The proposed algorithm contemplates the use of self-adaptive schemes of evolutionary operators, methods of Population Size Reduction (PSR), sampling techniques for the selection of individuals from the initial population, and stopping conditions that adapt to the needs of the optimization process. In addition, the developed algorithm contemplates the use of parallel computing strategies to accelerate the calculation of the values of the objective function.

Differential evolution is a stochastic, population-based algorithm developed for real-valued function optimization problems. It operates by having a population of individuals (x vector) that move around in the search space by recombining through crossover and mutation with other existing individuals in the population. Through a selection process, a newly generated individual is accepted as part of the population if the new individual x is an improvement; otherwise, it is discarded. This iteration process is repeated to find a vector x that optimizes a function f(x) [49]. As with other evolutionary algorithms, the search performance of differential evolution algorithms depends on the control parameter settings. A standard differential evolution has three main control parameters: population size, scaling factor F, and crossover factor CR. However, it is well-known that the optimal settings of these parameters are problem dependent. Therefore, when applying differential evolution to a real-world problem, it is often necessary to tune the control parameters to obtain the desired results. In practical cases, many researchers suggest the use of self-adaptive schemes to adjust online control parameters during the search process. A variant of differential evolution that applies this type of scheme is the success history-based adaptation for differential evolution (SHADE) [50]. In previous works by the authors of this article, the good performance of the SHADE algorithm for the optimization of aerodynamic bodies such as wings has been corroborated [51].

In SHADE, the mutation factor $F \in [0, 1]$ controls the magnitude of the differential mutation operator and $CR \in [0, 1]$ is the crossover factor. In the beginning, the contents of MCR, k, MF, k (k = 1, . . ., H) are all initialized to 0.5. In each generation g, the control parameters CRi and Fi used by each individual xi are generated by randomly selecting an index ri from [1, H], and then applying the formulas below:

$$CR_i = \begin{cases} 0 & \text{if } M_{CR,ri} = -1 \\ \text{randn}_i(M_{CR,ri}, 0.1) & \text{otherwise} \end{cases} \tag{43}$$

$$F_i = \text{randc}_i(M_{F,ri}, 0.1) \tag{44}$$

Here, $\text{randn}_i(M, 0.1)$, $\text{randc}_i(M, 0.1)$ are values selected randomly from normal and Cauchy distributions with mean M and variance 0.1. In case a value for CR_i outside of [0, 1] is generated, it is replaced by the limit value (0 or 1) closest to the generated value. When $F_i > 1$, F_i is truncated to 1, and when $F_i \leq 0$, Equation (44) is repeatedly applied to try to generate a valid value. In Equation (43), if $M_{CR,ri}$ has been assigned the "terminal value" -1, CR_i is set to 0.

The mutation strategy used by SHADE current-to-pbest/1 is a variant of the current-to-best/1 strategy where the greediness is adjustable using a parameter p.

$$v_{i,g} = x_{i,g} + F_i(x_{pbest,g} - x_{i,g}) + F_i(x_{r1,g} - x_{r2,g}) \tag{45}$$

The individual $x_{pbest,g}$ is randomly selected from the top p·NP members in the g-th generation ($p \in [0, 1]$). F_i is the F parameter used by individual x_i. The greediness of current-to-pbest/1 depends on the control parameter p to balance exploitation and exploration (small p behaves more greedily).

SHADE uses binary crossover strategy,

$$u_{j,i,g} = \begin{cases} v_{j,i,g} & \text{if } \text{rand}(0,1) \leq CR_i \\ x_{j,i,g} & \text{otherwise} \end{cases} \tag{46}$$

Perform selection between the trial vector ($u_{i,g}$) and target vector ($x_{i,g}$) using a greedy selection criterion,

$$x_{i,g+1} = \begin{cases} u_{i,g} & \text{iff } (u_{i,g}) \leq f(x_{i,g}) \\ x_{i,g} & \text{otherwise} \end{cases} \quad (47)$$

SHADE uses a historical memory M_{CR}, M_F which stores a set of CR, F values that have performed well in the past, and generates new CR, F pairs by directly sampling the parameter space close to one of these stored pairs.

In Algorithm 3, index k determines the position in memory to be updated. In generation g, the k-element in the memory is updated. At the beginning of the search, k is initialized to 1. k is incremented whenever a new element is inserted into the history. If k > H, k is set to 1. In the updated algorithm 1, note that when all individuals in generation g fail to generate a trial vector that is better than the parent, i.e., $S_{CR} = S_F = \emptyset$, the memory is not updated. The weighted Lehmer mean $mean_{WL}(S)$ is computed using the following formula:

$$mean_{WL}(S) = \frac{\sum_{m=1}^{|S|} w_m S_m^2}{\sum_{m=1}^{|S|} w_m S_m} \quad (48)$$

$$w_m = \frac{\Delta f_m}{\sum_{l=1}^{|S|} \Delta f_l} \quad (49)$$

$$\Delta f_m = \left| f(u_{m,g}) - f(x_{m,g}) \right| \quad (50)$$

Algorithm 3: Memory update algorithm in SHADE

Inputs: S_{CR}, S_F, $M_{CR,k,g}$, $M_{F,k,g}$, k, H
Outputs: $M_{CR,k,g+1}$, $M_{F,k,g+1}$, k
if $S_{CR} \neq \emptyset$ **and** $S_F \neq \emptyset$ **then**
    if $M_{CR,k,g} = -1$ **or** $\max(S_{CR}) = 0$ **then**
        $M_{CR,k,g+1} = -1$;
    else
        $M_{CR,k,g+1} = mean_{WL}(S_{CR})$;
    $M_{F,k,g+1} = mean_{WL}(S_F)$;
    k++;
    if k > H **then**
        k = 1;
else
    $M_{CR,k,g+1} = M_{CR,k,g}$;
    $M_{F,k,g+1} = M_{F,k,g}$;
return Outputs

The amount of fitness improvement Δf_m is used to influence the parameter adaptation (S refers to either S_{CR} or S_F). As M_{CR} is updated, if $M_{CR,k,g} = -1$ or $\max(S_{CR}) = 0$ (i.e., all elements of S_{CR} are 0), $M_{CR,k,g+1}$ is set to -1. Thus, if M_{CR} is assigned the terminal value -1, then M_{CR} will remain fixed at -1 until the end of the search. This has the effect of locking CR_i to 0 until the end of the search, causing the algorithm to enforce a "change-one-parameter-at-a-time" policy, which tends to slow down convergence, and is effective on multimodal problems.

The SHADE algorithm works well in conjunction with the PSR methods. To develop this optimization algorithm, we decided to incorporate the continuous adaptive population reduction (CAPR) method. The CAPR method gradually reduces the population size according to the change in the gradient of the fitness value [52].

$$NP_{g+1} = \begin{cases} \sqrt[\gamma]{\Delta_g / \Delta_{g-1}} & 0 < \Delta_g / \Delta_{g-1} < 1 \\ NP_g & \text{otherwise} \end{cases} \quad (51)$$

$$NP_{g+1} = \begin{cases} NP_{g+1} & NP_{g+1} > NP_{min} \\ NP_{min} & NP_{g+1} \leq NP_{min} \end{cases} \quad (52)$$

where

$$\Delta_g = \frac{f_{avg}(x_g) - f_{avg}(x_{g-1})}{f_{avg}(x_g)}, \Delta_{g-1} = \frac{f_{avg}(x_{g-1}) - f_{avg}(x_{g-2})}{f_{avg}(x_{g-1})} \quad (53)$$

In the third generation and in subsequent generations, the evaluated function values of all vectors in the population are averaged to be $f_{avg}(x_g)$. This value, together with that form the previous evaluation generation, is used to calculate the normalized gradient value Δ_g. Δ_{g-1} is calculated in a similar fashion using the previous average function evaluation value, $f_{avg}(x_{g-1})$, and the one before the previous $f_{avg}(x_{g-2})$. If the ratio Δ_g/Δ_{g-1} is within the range of [0, 1], then NP is reduced by a fraction equal to the γ-th root of the ratio Δ_g/Δ_{g-1}. The reason for taking root of the ratio is to slow down the population size reduction rate.

Another criterion to consider when increasing the performance of algorithms based on differential evolution is the generation of the initial population. It has been shown that a population, whose individuals are best distributed throughout the entire design space, has a greater chance of finding a global optimum, in addition to reducing the search time. The LHS design is a statistical method for generating a quasi-random sampling distribution. It is one of the most popular sampling techniques used in computer experiments because of its simplicity and projection properties with high-dimensional problems. The LHS is built as follows: each dimensional space, representing a variable, is cut into n sections, where n is the number of sampling points, and only one point is placed in each section [53].

For real-case optimization processes, it is common to use two types of stopping criteria. The following two types of stopping criteria were considered for this algorithm [54]:

- Exhaustion-based criteria: due to limited computational resources, the optimization run might be terminated after a certain number of objective function evaluations or CPU time. Typically, a maximum number of generations or number of objective function evaluations is used in combination with every stopping criterion to prevent the algorithm from running forever if a criterion cannot stop the run.
- Distribution-based criteria: for differential evolution algorithms, all individuals eventually converge to the optimum. Therefore, it can be concluded that convergence occurs when individuals are close to each other. Because it is assumed that the optimum is not known for the reference criterion, the distances between the population members are examined. This type of criterion can be applied to the design space or objective space.

One of the main disadvantages of evolutionary algorithms is that they need to evaluate multiple vectors to find the global optimum, which implies calculating the values of the objective function many times. One way to speed up the calculation process is to use parallel computing strategies. The strategy that was proposed to be used is Lightweight Pipelining (LP) [55]. The pipelining process provides an easy approach for downloading and using the models on demand. It helps in parallelization, which means that different jobs can be run in parallel. It also reduces redundancy and helps to inspect and debug the data flow in the model. Some features that pipeline provides are on-demand computing, tracking of data and computation, and inspection of data flow.

OpenVINT is the union and adaptation of each of the aforementioned algorithms and methods to achieve the objective of the optimization process mentioned at the beginning of this work. Coding of this algorithm was performed mainly on Python 3 in a GNU/LINUX environment.

3. Results

3.1. Case Studies

To evaluate the performance of the OpenVINT algorithm, two cases were performed:

- case 1—obtain the optimal design of a propeller used for an engine of a fixed-wing aircraft, considering a free flow velocity of 25 m/s and a minimum required thrust is 7.5 N;
- case 2—obtain the optimal design of a propeller used in a helicopter, considering a free flow velocity of 2 m/s and a minimum thrust required is 6.5 N.

In both cases, the following physical properties of the air flow were used: ρ, 1.225 kg/m^3; ν, 0.000014607 m^2/s; and a, 340.294 m/s. The intervals of the design variables implemented in each of the tests are shown in Table 1.

Table 1. Intervals of the design variables.

Parameter	Interval		Variable	Interval	
	Case 1	Case 2		Case 1	Case 2
$(c/d)_r$	[0.03, 0.10]	[0.05, 0.07]	x_{tt}	[0.30, 0.40]	[0.30, 0.40]
$(c/d)_m$	[0.03, 0.10]	[0.08, 0.13]	$\bar{r}_{xtm}$	[0.30, 0.80]	[0.20, 0.50]
$(c/d)_t$	[0.01, 0.02]	[0.01, 0.03]	y_{cr}	[0.01, 0.05]	[0.05, 0.08]
$\bar{r}_{cm}$	[0.35, 0.60]	[0.20, 0.50]	y_{cm}	[0.01, 0.05]	[0.05, 0.08]
α_r [°]	[−7, 7]	[0, 5]	y_{ct}	[0.005, 0.05]	[0.05, 0.08]
α_m [°]	[−5, 7]	[0, 5]	$\bar{r}_{ycm}$	[0.30, 0.80]	[0.20, 0.50]
α_t [°]	[−5, 7]	[0, 5]	x_{cr}	[0.30, 0.40]	[0.30, 0.40]
$\bar{r}_{\alpha m}$	[0.25, 0.75]	[0.20, 0.50]	x_{cm}	[0.30, 0.45]	[0.30, 0.40]
y_{tr}	[0.12, 0.20]	[0.10, 0.20]	x_{ct}	[0.30, 0.45]	[0.30, 0.40]
y_{tm}	[0.12, 0.16]	[0.08, 0.10]	$\bar{r}_{xcm}$	[0.30, 0.80]	[0.20, 0.50]
y_{tt}	[0.10, 0.12]	[0.08, 0.10]	n_m [rev/min]	$[5 \times 10^3, 10 \times 10^3]$	$[5 \times 10^3, 10 \times 10^3]$
$\bar{r}_{ytm}$	[0.30, 0.80]	[0.20, 0.50]	B	[2, 4]	[2, 3]
x_{tr}	[0.30, 0.40]	[0.30, 0.40]	d [m]	[0.3, 0.3]	[0.254, 0.254]
x_{tm}	[0.30, 0.45]	[0.30, 0.40]			

For both optimization cases the following parameters of the optimization algorithm were taken:

- in real optimization problems it is considered to use at least 50 individuals in the initial population [56], and as a minimum population 10 individuals were considered;
- the stop conditions contemplated were a maximum number of evaluated generations of 200, and an ε value of 1 W to fulfill the condition indicated in line 45 of Algorithm 4;
- a γ factor of 50 was used in (51);
- finally, an initial U^* value of 350 W was considered.

All the tests were performed on a PC whose characteristics are as follows:

- operative system, Ubuntu 22.04.3 LTS;
- processor, 13th Gen Intel® Core™ i5-13400Fx12;
- RAM memory, 64 GB;

Algorithm 4: OpenVINT algorithm

Inputs: d, V_∞, ρ, ν, a, T_{min}, $\bar{r}$, NS, $[X_{min}, X_{max}]$, G, NP, NP_{min}, ε, U^*, γ, H, p
Outputs: x_{opt}, $L(x_{opt})$
//Initialization phase
1 g = 1;
2 Initialize of metrics;
3 Initialize population P_g with LHS;
//Parallelized loop by joblib
4 **for** i = 1 **to** NP **do**
5 Apply Algorithm 1 for $x_{i,g}$;
//Parallelized loop by joblib
6 **for** i = 1 **to** NP **do**
7 Get $T(x_{i,g})$, $W(x_{i,g})$, $\eta_d(x_{i,g})$, $\eta_s(x_{i,g})$ with Algorithm 2;
8 Get $\psi(x_g)$ with (2) and $L(x_g)$ with (1);
9 Update U^*;
10 Save $L_{avg}(x_g)$;
11 Save data of generation g;
12 Set all values in M_{CR}, M_F to 0.5;
13 Archive A = ∅;

Algorithm 4: *Cont.*

14 k = 1;
//Main loop
15 **for** g = 2 **to** G **do**
16 $S_{CR} = \emptyset$, $S_F = \emptyset$;
17 **for** i = 1 **to** NP **do**
18 r_i = select from [1, H] randomly;
19 Get $CR_{i,g}$ with (43);
20 Get $F_{i,g}$ with (44);
21 Get mutation vector $v_{i,g}$ with (45);
22 Get trial vector $u_{i,g}$ with (46);
//Parallelized loop by joblib
23 **for** i = 1 **to** NP **do**
24 Apply Algorithm 1 with $u_{i,g}$;
//Parallelized loop by joblib
25 **for** i = 1 **to** NP **do**
26 Get $T(u_{i,g})$, $W(u_{i,g})$, $\eta_d(u_{i,g})$, $\eta_s(u_{i,g})$ with Algorithm 2;
27 Get $\psi(u_g)$ with (2) and $L(u_g)$ with (1);
28 Update U^*;
29 **for** i = 1 **to** NP **do**
30 **if** $L(u_{i,g}) \leq L(x_{i,g})$ **then**
31 $x_{i,g+1} = u_{i,g}$;
32 $x_{i,g} \rightarrow A$;
33 $CR_{i,g} \rightarrow S_{CR}$, $F_{i,g} \rightarrow S_F$;
34 **else**
35 $x_{i,g+1} = x_{i,g}$;
36 Update memories M_{CR} and M_F with Algorithm 3;
37 Save $L_{avg}(x_{g+1})$;
38 **if** $g \geq 3$ **then**
39 Get Δ_g and Δ_{g-1} with (53);
40 Get NP_{g+1} with (51);
41 **if** $NP_{g+1} < NP_{min}$ **then**
42 Apply (52);
43 $(NP_g - NP_{g+1})$-th worst vectors $\rightarrow$ A;
44 Save data of generation g+1;
45 **if** $|L_{avg}(x_{g+1}) - L_{opt}(x_{g+1})| \leq \varepsilon$ **then**
46 **break**;
47 k++;
48 Print metrics plots;
49 Output
50 Drawing the optimal propeller in point clouds;

3.2. Optimization Results

For case 1, the following parameters of the optimization algorithm were taken:

- 6 tests were performed, 3 of them were with an initial population of 50 vectors and the other 3 with an initial population of 100 vectors;
- a minimum population of 10 vectors was considered in all tests;
- the stop conditions contemplated were a maximum number of evaluated generations of 200, and an ε value of 1 W to fulfill the condition indicated in line 48 of algorithm 4;
- a γ factor of 50 was used in (51);
- finally, an initial U^* value of 350 W was considered.

Figure 9 shows the convergence graphs of the metrics used to evaluate the propeller optimization process for case 1.

Table 2 shows the values of the optimal design vector as well as the optimal values of the required power output, thrust provided, and the dynamic efficiency of the propeller. In addition, the average values, and the coefficient of variation (CV) for each parameter are shown, which allows evaluation of the homogeneity of the data.

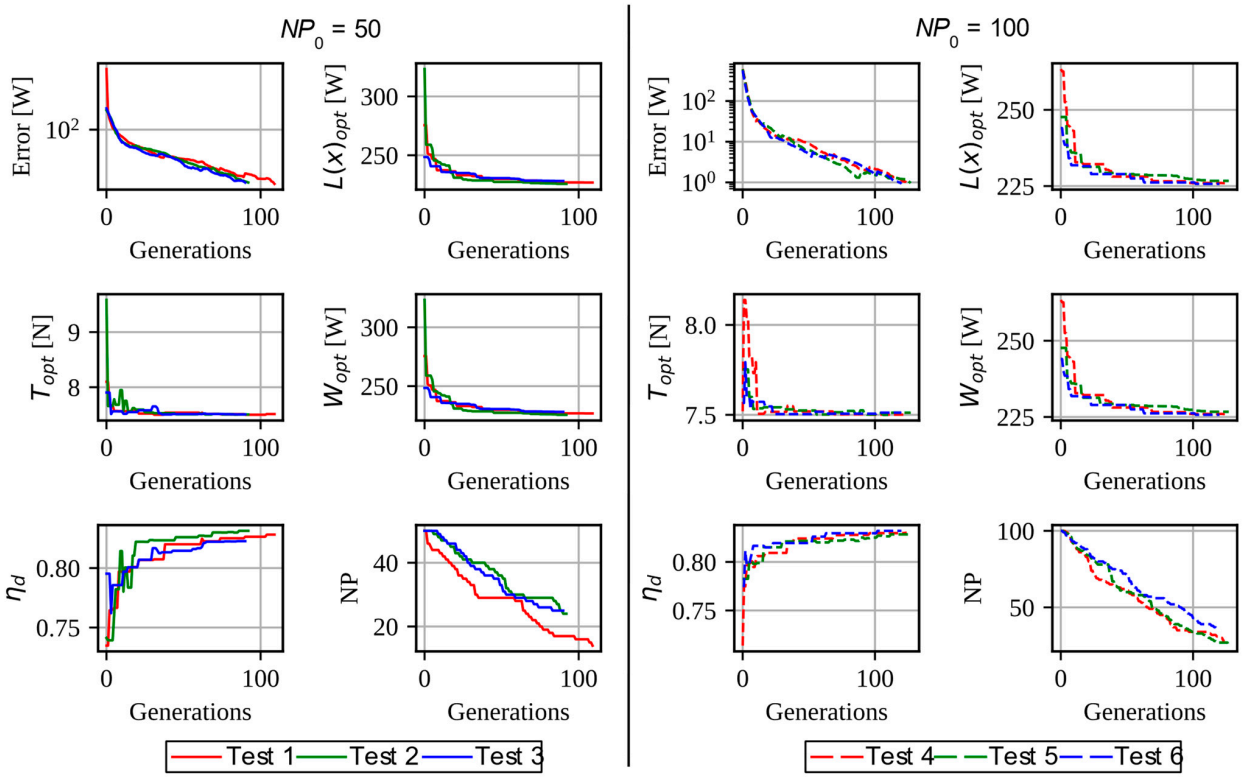


Figure 9. Convergence of the metrics in the tests corresponding to case 1.

Table 2. Results obtained from case 1.

Parameter	NP = 50					NP = 100				
	Test 1	**Test 2**	**Test 3**	**Mean**	**CV, [%]**	**Test 4**	**Test 5**	**Test 6**	**Mean**	**CV, [%]**
$(c/d)_r$	0.043	0.082	0.094	0.073	29.45	0.035	0.038	0.043	0.039	8.04
$(c/d)_m$	0.100	0.094	0.088	0.094	5.21	0.085	0.087	0.100	0.091	7.40
$(c/d)_t$	0.012	0.015	0.018	0.015	15.33	0.020	0.020	0.019	0.020	3.42
$\bar{r}_{cm}$	0.509	0.400	0.600	0.503	16.29	0.551	0.587	0.522	0.553	4.82
α_r [°]	0.243	−0.784	6.288	1.916	162.84	4.214	−7.000	6.245	1.153	505.15
α_m [°]	6.144	5.278	4.307	5.243	14.31	6.956	6.081	5.609	6.216	8.98
α_t [°]	4.823	6.973	2.324	4.707	40.36	4.632	4.770	5.217	4.873	5.12
$\bar{r}_{\alpha m}$	0.253	0.251	0.250	0.251	0.60	0.253	0.286	0.605	0.381	41.62
y_{tr}	0.070	0.060	0.066	0.066	6.41	0.060	0.061	0.061	0.061	0.75
y_{tm}	0.060	0.061	0.061	0.061	0.66	0.060	0.060	0.060	0.060	0.00
y_{tt}	0.059	0.060	0.054	0.057	3.99	0.053	0.060	0.051	0.055	6.71
$\bar{r}_{ytm}$	0.745	0.300	0.300	0.448	46.74	0.598	0.530	0.364	0.497	19.73
x_{tr}	0.327	0.300	0.309	0.312	3.52	0.333	0.320	0.308	0.320	3.21
x_{tm}	0.329	0.335	0.300	0.321	4.73	0.332	0.336	0.315	0.328	2.73
x_{tt}	0.330	0.315	0.306	0.317	3.15	0.339	0.384	0.345	0.356	5.65
$\bar{r}_{xtm}$	0.787	0.722	0.300	0.603	35.75	0.300	0.300	0.357	0.319	8.35
y_{cr}	0.050	0.010	0.029	0.030	54.70	0.026	0.047	0.011	0.028	52.36
y_{cm}	0.010	0.011	0.014	0.012	15.13	0.010	0.010	0.010	0.010	0.00
y_{ct}	0.005	0.016	0.028	0.016	57.93	0.006	0.005	0.006	0.006	8.31
$\bar{r}_{ycm}$	0.358	0.459	0.652	0.490	24.90	0.483	0.358	0.712	0.518	28.33
x_{cr}	0.338	0.399	0.331	0.356	8.65	0.398	0.395	0.364	0.386	3.95
x_{cm}	0.443	0.443	0.306	0.397	16.28	0.352	0.386	0.387	0.37	4.33
x_{ct}	0.361	0.441	0.310	0.370	14.59	0.399	0.367	0.422	0.396	5.77
$\bar{r}_{xcm}$	0.692	0.407	0.778	0.626	25.32	0.643	0.800	0.505	0.649	18.58
n_m [rev/min]	6156	6238	6765	6386	4.22	6281	6312	5993	6195	2.32
B	2	2	2	2	0	2	2	2	2	0
d [m]	0.300	0.300	0.300	0.300	0	0.300	0.300	0.300	0.300	0
W [W]	226.8	225.7	228.1	226.9	0.44	226.0	226.7	225.7	226.1	0.18
T [N]	7.513	7.504	7.506	7.508	0.05	7.501	7.512	7.512	7.509	0.07
η_d	0.828	0.831	0.823	0.827	0.41	0.830	0.828	0.832	0.830	0.18
Computation time [s]	3897	4728	4417	4347	7.89	9987	9526	10,703	10,072	4.81

The geometric differences between each of the blades obtained for case 1 can be visualized in Figures 10 and 11.

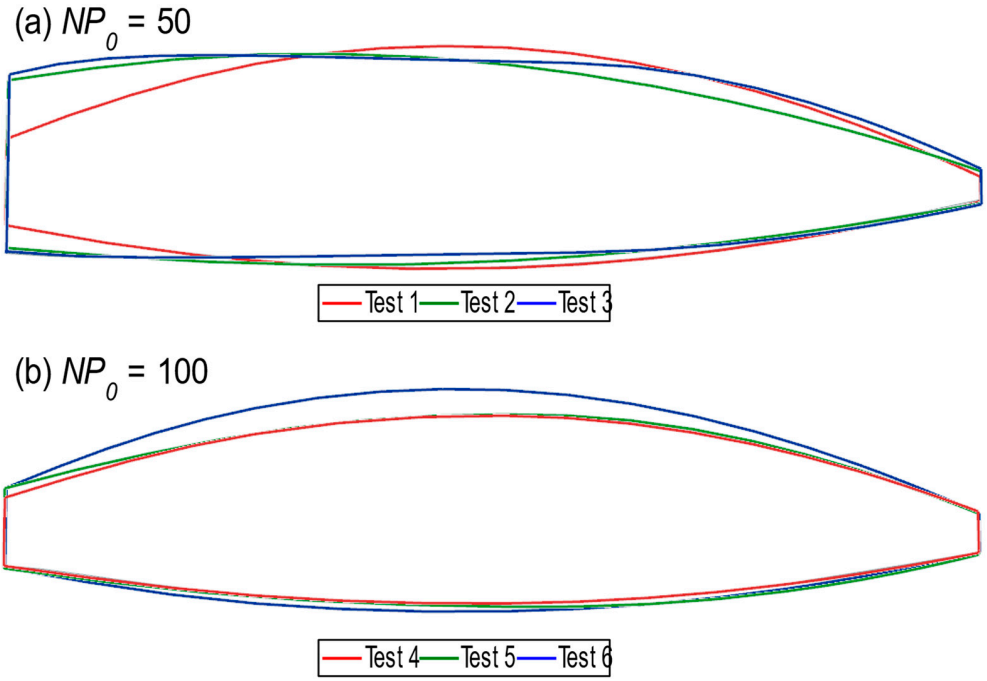


Figure 10. Top view of the propeller blades obtained for case 1.

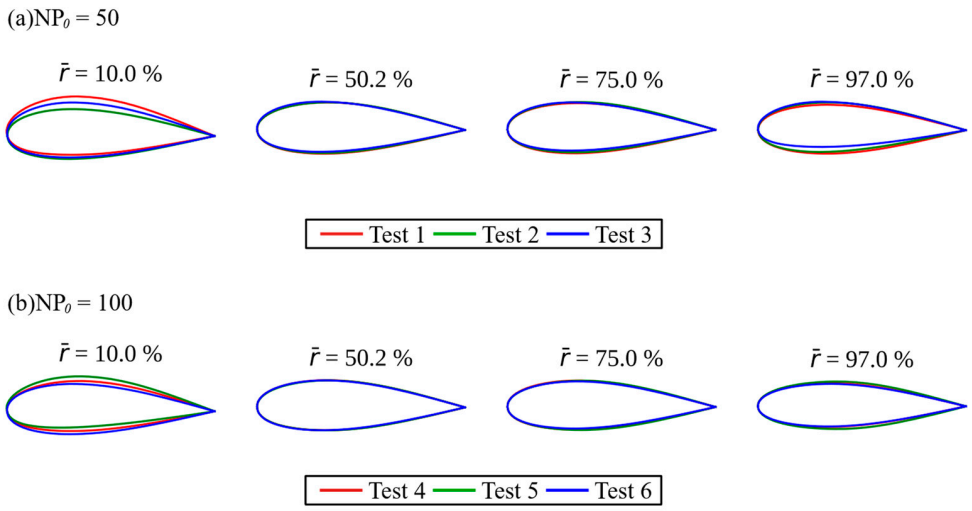


Figure 11. Examples of the cross-sections of the blades obtained for case 1. The airfoils are normalized with chord of length 1.

For case 2 the following parameters of the optimization algorithm were taken:

- A total of 3 tests were performed, with an initial population of 50 vectors and a minimum population of 10 vectors;
- the stop conditions contemplated were a maximum number of evaluated generations of 200, and an ε value of 1 W to fulfill the condition indicated in line 48 of algorithm 4;
- a γ factor of 50 was used in (51);
- finally, an initial U^* value of 350 W was considered.

Figure 12 shows the convergence graphs of the metrics used to evaluate the propeller optimization process. The final values of the optimization are shown in more detail in Table 3.

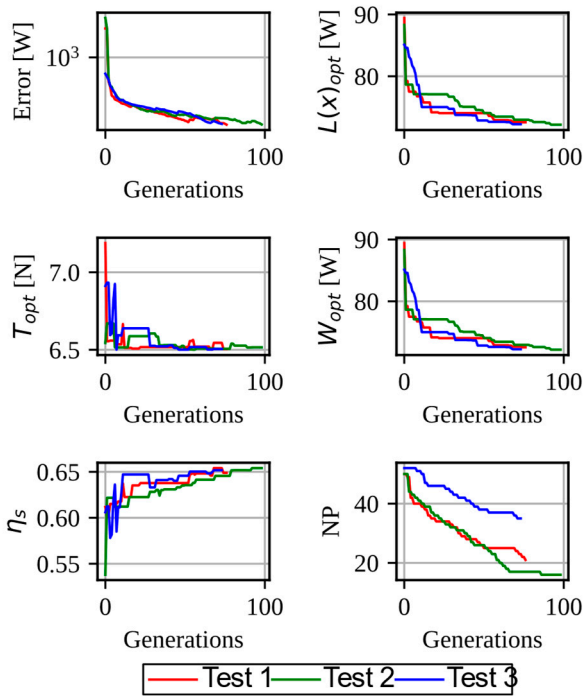


Figure 12. Convergence of the metrics in the tests corresponding to case 2.

Table 3. Results obtained from the case 2.

Parameter	Test 1	Test 2	Test 3	Mean	CV, [%]
$(c/d)_r$	0.063	0.070	0.055	0.063	9.78
$(c/d)_m$	0.130	0.120	0.125	0.125	3.27
$(c/d)_t$	0.030	0.028	0.030	0.029	3.21
$\bar{r}_{cm}$	0.456	0.500	0.478	0.478	3.76
α_r [°]	0.000	0.718	3.579	1.432	107.93
α_m [°]	4.625	2.254	3.772	3.550	27.62
α_t [°]	1.619	1.193	3.734	2.182	50.92
$\bar{r}_{\alpha m}$	0.435	0.327	0.500	0.421	16.96
y_{tr}	0.063	0.073	0.093	0.076	16.34
y_{tm}	0.042	0.040	0.040	0.041	2.31
y_{tt}	0.046	0.043	0.042	0.044	3.89
$\bar{r}_{ytm}$	0.251	0.200	0.208	0.220	10.19
x_{tr}	0.300	0.382	0.356	0.346	9.89
x_{tm}	0.331	0.301	0.329	0.320	4.28
x_{tt}	0.361	0.300	0.390	0.350	10.71
$\bar{r}_{xtm}$	0.497	0.200	0.248	0.315	41.33
y_{cr}	0.064	0.062	0.069	0.065	4.53
y_{cm}	0.051	0.051	0.052	0.051	0.92
y_{ct}	0.050	0.053	0.051	0.051	2.43
$\bar{r}_{ycm}$	0.272	0.226	0.222	0.240	9.45
x_{cr}	0.386	0.398	0.334	0.372	7.45
x_{cm}	0.300	0.302	0.300	0.301	0.31
x_{ct}	0.323	0.301	0.301	0.308	3.36
$\bar{r}_{xcm}$	0.263	0.249	0.209	0.240	9.52
n_m [rev/min]	6720	7662	6705	7209	6.37
B	2	2	2	2	0.00
d [m]	0.254	0.254	0.254	0.254	0.00
W [W]	72.56	72.17	72.24	72.32	0.23
T [N]	6.505	6.516	6.505	6.509	0.08
η_s	0.649	0.654	0.652	0.652	0.32
Computation time [s]	3333	3743	3227	3434	6.48

Graphically, the differences between the blades obtained in each of the tests can be observed in Figures 13 and 14.

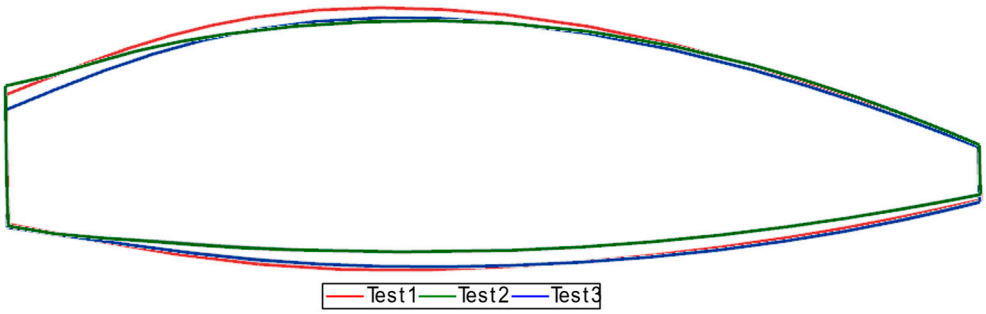


Figure 13. Top view of the propeller blades obtained for case 2.

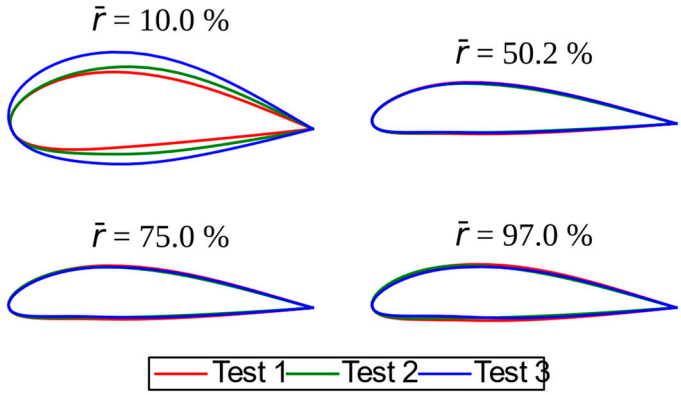


Figure 14. Examples of the cross-sections (airfoils) of the blades obtained for case 2. The airfoils are normalized with chord of length 1.

3.3. Experimental Validation

To verify the results obtained in the optimization process, one of the propellers obtained in the tests performed for case 2 (the propeller obtained in test 3) was selected. The propeller was manufactured by contact molding continuous carbon fibers using epoxy thermosetting binder in a closed mold. The mold was fabricated using a CNC milling machine from the geometry obtained in test 3 (Figure 15). The outer layers and the spars passing through the sleeve comprise unidirectional fibers, and the inner two layers are stacked from plain fabric with fiber orientation $\pm 45^{\circ}$.

(**a**) (**b**)

Figure 15. (**a**) Injection mold for the manufacture of the propellers; (**b**) manufactured propellers.

The proposed experimental setup allows the monitoring of the angular velocity, thrust, and torque of the propeller. The angular velocity of the propeller was monitored using

two different methods: measurement from the electric motor with a PWM frequency meter and measurement from the propeller with a laser photo tachometer. The thrust and torque of the propeller are measured using strain gauges at the base of the motor. The power requirement of the propeller was calculated as the product of torque and angular velocity (see Figure 16) [57].

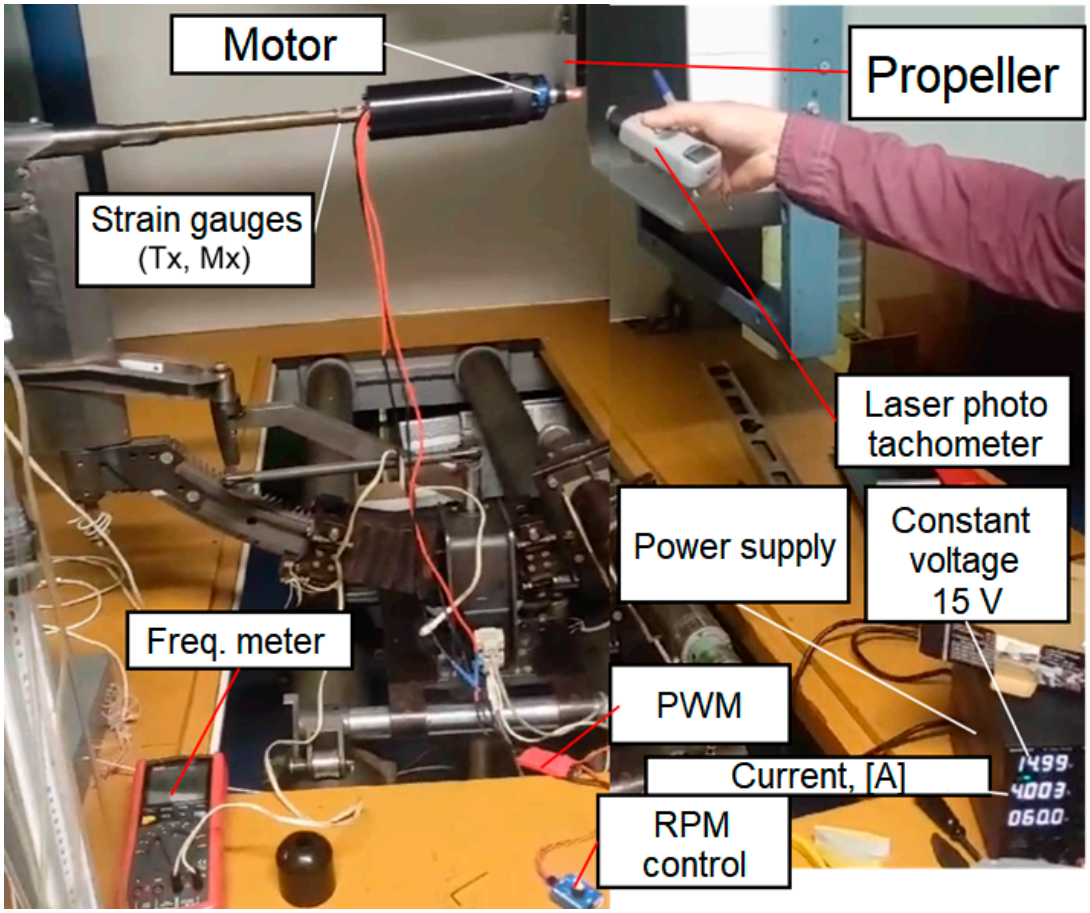


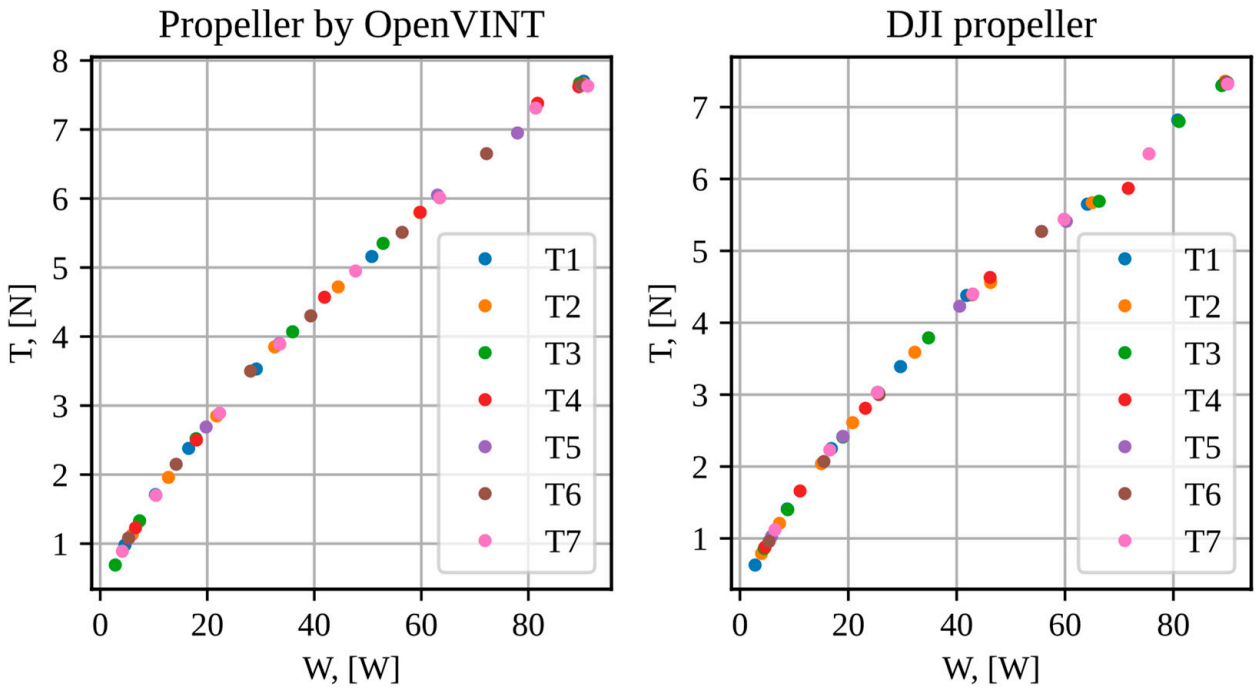
Figure 16. Experimental setup.

A comparison was made of the dependences of thrust on the required power for two propellers—designed (test 2, case 3) and commercial DJI 10 × 4.5″ propeller (Figure 17). Seven experimental tests (T1. . .T7) were carried out for each propeller, varying in rotational speed (Figure 18). Each test consists of from 6 to 8 measurements—a total of 46 measurements for designed propeller and 49 measurements for DJI propeller. The dependences of thrust on required power for each test were interpolated by smooth univariable spline of 3 degrees and then lines of average values and scatter equal to the standard deviation are plotted (Figure 19).

Comparison of calculated by ISM and experimental data for the designed propeller (Figure 19a) showed that for the main propeller regimes (trust more than 2N) the difference does not exceed 3.5%. A comparison between the designed propeller and a commercially available propeller (DJI 10 × 4.5″ propeller) is shown in Figure 19b. The proposed algorithm demonstrates that it can design a propeller that provides a higher thrust (from 10 to 15%) with the same power requirement as commercial propellers in main operating mode.

(**a**) (**b**)

Figure 17. The designed (**a**) and commercial (**b**) propellers installed in the experimental setup.



(**a**) (**b**)

Figure 18. Experimental data of thrust from required power dependence for propellers: (**a**) designed, (**b**) commercial DJI propeller.

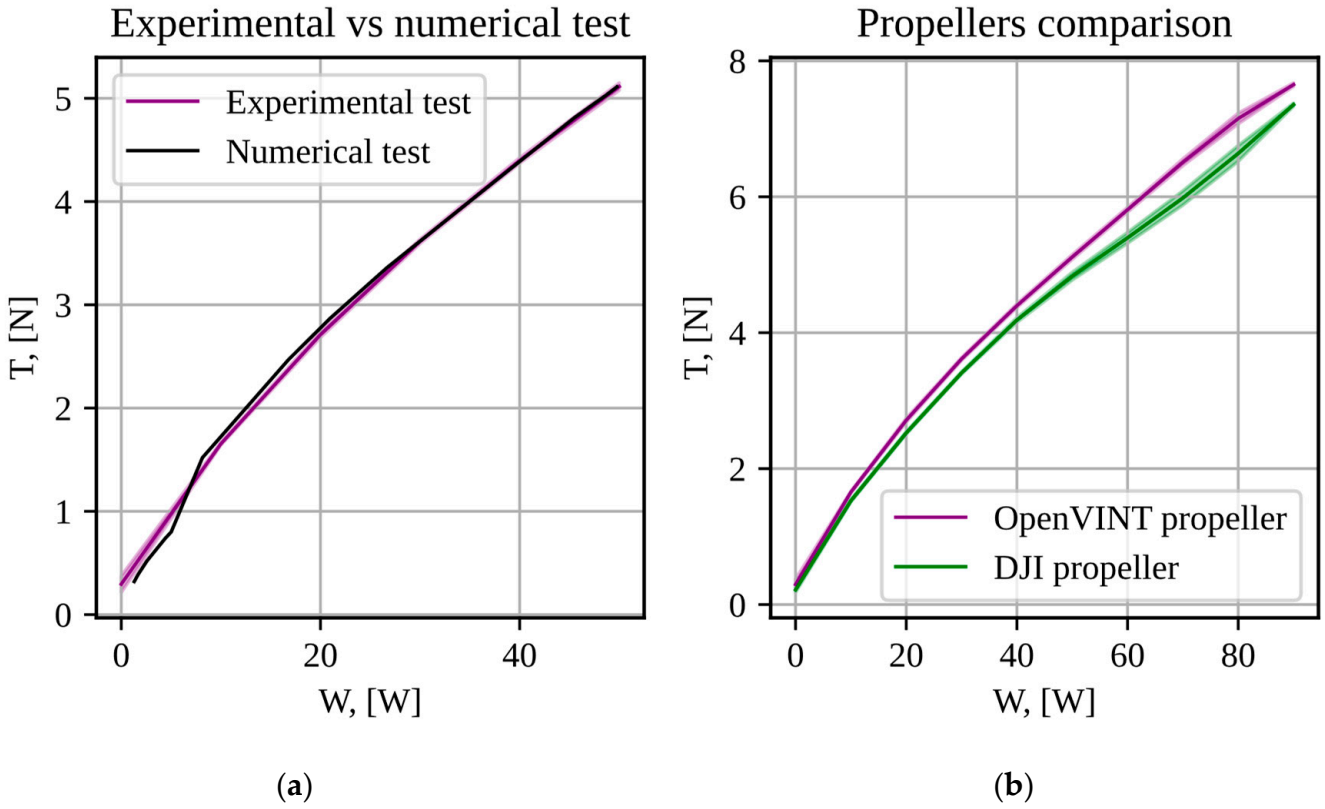


Figure 19. Mean measurement values and scatter of propeller thrust from power requirement: (**a**) comparison for designed propeller between experimental values and numerical test, calculated by ISM, (**b**) comparison between experimental values for designed and commercial DJI propeller.

4. Discussion

In the results of the tests carried out, presented in Tables 2 and 3, the following points could be observed:

- for the optimization of a helicopter propeller, it is sufficient to use an initial population of 50 vectors to obtain similar results (CV values less than 30%);
- to optimize a tractor propeller, it was necessary to use an initial population of 100 vectors to improve the search for a global optimal;
- in both cases, there is a greater variation in the parameters that describe the geometry of the blade root, whereas in the parameters that describe the rest of the blade geometry, the results were more homogeneous.

Figures 9 and 12 illustrate that in the most representative blade section (in 75% of the blade length), practically similar cross-sectional (airfoil) geometries are obtained. Similar aerodynamic behavior implies that similar values of thrust provided and thrust required are obtained.

When performing the experimental tests with one of the propellers obtained in case 2, it was observed that the aerodynamic characteristics of the propeller obtained with the help of the ISM and XFOIL were comparable to those obtained in the experimental tests, as shown in Figure 19a. These results provide certainty in the calculated values of the objective function for each vector evaluated in the optimization process. While in the comparison that was made with one of the optimized propellers and with a commercial propeller, it was observed that the OpenVINT algorithm provides new propeller options that require less power to achieve a certain thrust.

Regarding the calculation speed of the algorithm, it can be observed that the OpenVINT algorithm provides better performance when optimizing propellers used in helicopters. For this type of propeller, the algorithm requires approximately 1 h of calculation

(using a computer as described in Section 3.1), whereas optimizing a tractor propeller for the engine of a fixed-wing aircraft requires approximately 3 h of calculation. Of course, these times can still improve the design space for each type of propeller. This type of analysis is proposed for future studies.

5. Conclusions

The paper presents an algorithm for propeller shape optimization based on isolated section method, similar to BEMT. Using Bezier curves to specify the dependencies of the parameters of thickness, curvature, and positions of the maximum thickness and maximum curvature of the airfoil along the blade span allows the aerodynamic twist of the blade to be varied.

The usage of a cross-section airfoil angle of attack as a function of blade span as design variable adds robustness to the optimization algorithm, protecting from calculation airfoil characteristics at supercritical angles of attack and reduce the number of calculations during the optimization. This makes it possible to reduce the optimization time and increase the accuracy of the optimization solution with geometric models with a large number of design variables and cross-sections used for calculations. Interpolation of the aerodynamic characteristics of sections along the blade span also makes it possible to reduce calculation time. The proposed algorithm allows for obtaining the optimal geometry of the pusher and tractor propellers based on 27 design variables, containing a variable along the blade span airfoil, a nonlinear distribution of twist and chord along the span in three hours of calculation on a personal computer.

In the tests to which OpenVINT optimization algorithm was subjected, it was observed that the parameterization of the proposed propeller geometry is good but still has room for improvement. From the 27 proposed design parameters, the parameters describing the shape of the blade root have little importance in determining the aerodynamic performance of the propeller, which implies that the number of design parameters can be reduced, or the value of these parameters can be refined using multidisciplinary optimization, for example based on stiffness and strength constraints.

The experimental tests validate the calculation method and confirm that the algorithm in general can provide configurations with better aerodynamic performance to the propellers that are available on the market.

Author Contributions: Conceptualization, D.N., E.K. (Evgenii Kurkin) and O.L.; methodology, D.N., J.G.Q.-P., E.K. (Evgenii Kurkin) and O.L.; software, J.G.Q.-P. and E.K. (Evgenii Kurkin); validation, D.N. and J.G.Q.-P.; formal analysis, V.C., E.K. (Ekaterina Kurkina) and V.H.H.; resources, A.S.; writing—original draft preparation, A.S., J.G.Q.-P., O.L. and V.C.; writing—review and editing, J.G.Q.-P., E.K. (Evgenii Kurkin), O.L., D.N., V.C., E.K. (Ekaterina Kurkina), V.H.H. and A.S.; visualization, V.C., E.K. (Ekaterina Kurkina) and V.H.H.; supervision, D.N.; project administration, D.N.; funding acquisition, D.N., O.L. and A.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Samara National Research University Development Program for 2021–2030 as part of the Strategic Academic Leadership Program "Priority 2030", grant number PR-NU 2.1-08-2023.

Data Availability Statement: The raw data cannot be shared at this time as the data also forms part of an ongoing study.

Acknowledgments: We would like to express our gratitude to Elena Tarasova and Damian Josue Guerra Guerra for the aerodynamic experiment assistance and Georgy Charkviani and Dmitry Kashirsky for help with propellers manufacturing.

Conflicts of Interest: The authors declare no conflicts of interest. The founders had no role in the design of the study; in the collection, analysis, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Chen, G.; Ma, D.; Jia, Y.; Xia, X.; He, C. Comprehensive Optimization of the Unmanned Tilt-Wing Cargo Aircraft with Distributed Propulsors. *IEEE Access* **2020**, *8*, 137867–137883. [CrossRef]
2. Traub, L.W. Considerations in optimal propeller design. *J. Aircr.* **2021**, *58*, 8. [CrossRef]
3. Betz, A. *Introduction to the Theory of Flow Machines*; Pergamon Press: Oxford, UK, 1966.
4. Betz, A. *Schraubenpropeller Mit Geringstem Energieverlust (Screw Propeller with Least Energy Loss)*; Akademie der Wissenschaften: Gottingen, Germany, 1919; pp. 193–217.
5. Adkins, C.N.; Liebeck, R.H. Design of Optimum Propellers. *J. Propul. Power.* **1994**, *10*, 676–682. [CrossRef]
6. Aleksandrov, V.L. *Vozdushnye Vinty*; Gosudarstvennoe Izdatel'stvo Oboronnoj Promyshlennosti: Moscow, Russia, 1951; 476p. (In Russian)
7. Kümmel, A.; Breitsamter, C. Multi-Disciplinary Framework for Propeller Blade Design. *IOP Conf. Ser. Mater. Sci. Eng.* **2021**, *1024*, 012060. [CrossRef]
8. Chen, W.; Chiu, K.; Fuge, M.D. Airfoil Design Parameterization and Optimization Using Bézier Generative Adversarial Networks. *AIAA J.* **2020**, *58*, 4723–4735. [CrossRef]
9. Gur, O.; Rosen, A. Optimization of Propeller Bases Propulsion Systems. *J. Aircr.* **2009**, *46*, 95–106. [CrossRef]
10. Dorfling, J.; Rokhsaz, K. Constrained and Unconstrained Propeller Blade Optimization. *J. Aircr.* **2015**, *52*, 1179–1188. [CrossRef]
11. Yu, P.; Peng, J.; Bai, J.; Han, X.; Song, X. Aeroacoustic and Aerodynamic Optimization of Propeller Blades. *Chin. J. Aeronaut.* **2020**, *33*, 826–839. [CrossRef]
12. Wang, S.; Zhang, S.; Ma, S. An Energy Efficiency Optimization Method for Fixed Pitch Propeller Electric Aircraft Propulsion Systems. *IEEE Access* **2019**, *7*, 159986–159993. [CrossRef]
13. Bekele, E.G.; Nicklow, J.W. Multi-objective automatic calibration of SWAT using NSGA-II. *J. Hydrol.* **2007**, *341*, 165–176. [CrossRef]
14. Ma, R.; Zhong, B.W.; Liu, P.Q. Optimization design study of low-Reynolds-number high-lift airfoils for the high-efficiency propeller of low-dynamic vehicles in stratosphere. *Sci. China Technol. Sci.* **2010**, *53*, 2792–2807. [CrossRef]
15. Slavik, S.; Klesa, J.; Brabec, J. Propeller Selection by Means of Pareto-Optimal Sets Applied to Flight Performance. *Aerospace* **2020**, *7*, 21. [CrossRef]
16. Fang, B.R. *Design of Aircraft Aerodynamic Configuration*; Chinese Aviation Industry Press: Beijing, China, 1997; pp. 97–130. [CrossRef]
17. Colozza, A. *High Altitude Propeller Design and Analysis Overview*; NASA/CR 98-208520; Federal Data Systems: Cleveland, OH, USA, 1998.
18. Morgado, J.; Abdollahzadeh, M.; Silvestre, M.; Páscoa, J. High Altitude Propeller Design and Analysis. *Aerosp. Sci. Technol.* **2015**, *45*, 398–407. [CrossRef]
19. Burdett, T.A.; Van Treuren, K.W. A Theoretical and Experimental Comparison of Optimizing Angle of Twist Using BET and BEMT. In Proceedings of the ASME Turbo Expo 2012: Turbine Technical Conference and Exposition. Volume 6: Oil and Gas Applications; Concentrating Solar Power Plants; Steam Turbines; Wind Energy, Copenhagen, Denmark, 11–15 June 2012; pp. 797–809. [CrossRef]
20. Oliveira, H.A.; de Matos, J.G.; Ribeiro, L.A.d.S.; Saavedra, O.R.; Vaz, J.R.P. Assessment of Correction Methods Applied to BEMT for Predicting Performance of Horizontal-Axis Wind Turbines. *Sustainability* **2023**, *15*, 7021. [CrossRef]
21. Goldstein, S. On The Vortex Theory of Screw Propellers. *Proc. R. Soc.* **1929**, *123*, 440–465. [CrossRef]
22. Zhong, W.; Wang, T.G.; Zhu, W.J.; Shen, W.Z. Evaluation of Tip Loss Corrections to AD/NS Simulations of Wind Turbine Aerodynamic Performance. *App. Sci.* **2019**, *9*, 4919. [CrossRef]
23. Branlard, E.; Dixon, K.; Gaunaa, M. Vortex Methods to Answer The Need For Improved Understanding And Modelling of Tip-Loss Factors. *IET Renew. Power Gener.* **2013**, *7*, 311–320. [CrossRef]
24. Elkhchine, Y.; Sriti, M. Tip Loss Factor Effects on Aerodynamic Performances of Horizontal Axis Wind Turbine. *Energy Procedia* **2017**, *118*, 136–140. [CrossRef]
25. Shaidakov, V.I. *Aerodinamika Vinta v kol'ce: Uch Posobie*; MAI: Moscow, Russia, 1996; 88p. (In Russian)
26. Wald, Q.R. The Aerodynamics of Propellers. *Prog. Aerosp. Sci.* **2006**, *42*, 85–128. [CrossRef]
27. XFOIL Subsonic Airfoil Development System. Available online: https://web.mit.edu/drela/Public/web/xfoil/ (accessed on 10 January 2024).
28. Munguia, J.; Van Treuren, W. *Designing Small Propellers for Optimum Efficiency*; Baylor University: Waco, TX, USA, 2019.
29. Pierret, S.S.; Van den Braembussche, R.A. Turbomachinery Blade Design Using a Navier-Stokes Solver and Artificial Neural Net-work. *J. Turbomach.* **1999**, *121*, 326–332. [CrossRef]
30. Segui, M.M.; Castelar, Y.; Botez, R.M. Wing Airfoils Generation Based on a New Parametric Curve for Aerodynamic Optimization Application. In Proceedings of the CASI AERO-2019 Conference, Big Sky, MT, USA, 2–9 March 2019; Canadian Aeronautics and Space Institute: Montreal, QC, Canada, 2019.
31. Shikhar Jaiswal, A. Shape Parameterization of Airfoil Shapes Using Bezier Curves. In *Innovative Design and Development Practices in Aerospace and Automotive Engineering*; Bajpai, R.P., Chandrasekhar, U., Eds.; Lecture Notes in Mechanical Engineering; Springer: Berlin/Heidelberg, Germany, 2017. [CrossRef]
32. Alexeev, R.A.; Tishchenko, V.A.; Gribin, V.G.; Gavrilov, I.Y. Turbine Blade Profile Design Method Based on Bezier Curves. *J. Phys. Conf. Ser.* **2017**, *891*, 012254. [CrossRef]

33. Hao, X.; Zhang, W.; Liu, X.; Liu, J. Aerodynamic and Aeroacoustic Optimization of Wind Turbine Blade by a Genetic Algorithm. In Proceedings of the 46th AIAA Aerospace Sciences Meeting and Exhibit,AIAA 2008-1331, Reno, NV, USA, 7–10 January 2008. [CrossRef]
34. Borovkov, A.I.; Voinov, I.B.; Ibraev, D.F. Determination of the Optimal Aerodynamic Shape for a Propeller Blade Based on Parametric Optimization. *Russ. Aeronaut. (Iz VUZ)* **2021**, *64*, 173–180. [CrossRef]
35. Ayaz Ümütlü, H.C.; Kiral, Z. Airfoil Shape Optimization using Bézier Curve and Genetic Algorithm. *Aviation* **2022**, *26*, 32–40. [CrossRef]
36. Xin, P.; Dawei, L.; Jixiang, S.; Yonghong, L. Airfoil Aerodynamic Optimization Based on an Improved Genetic Algorithm. In Proceedings of the Fifth International Conference on Intelligent Systems Design and Engineering Applications, Hunan, China, 15–16 June 2014; pp. 133–137. [CrossRef]
37. Jiao, J.; Song, B.-F.; Zhang, Y.-G.; Li, Y.-B. Optimal Design and Experiment of Propellers for High Altitude Airship. *Proc. Inst. Mech. Eng. Part. G J. Aerosp. Eng.* **2017**, *232*, 1887–1902. [CrossRef]
38. Bocii, L.S.; Di Noia, L.P.; Rizzo, R. Optimization of the Energy Storage of Series-Hybrid Propelled Aircraft by means of Integer Differential Evolution. *Aerospace* **2019**, *6*, 59. [CrossRef]
39. Muratoglu, A.; Tekin, R.; Ertugrul, O.F. Hydrodynamic Optimization of High-Performance Blade Sections for Stall Regulated Hydrokinetic Turbines Using Differential Evolution Algorithm. *Ocean Eng.* **2021**, *220*, 108389. [CrossRef]
40. Spera, D.A. *Models of Lift and Drag Coefficients of Stalled and Unstalled Airfoils in Wind Turbines and Wind Tunnels*; Technical Report NASA/CR-2008-215434; GRC: Cleveland, OH, USA, 2008.
41. Quijada Pioquinto, J.G.; Kurkin, E.I.; Nazarov, D.V.; Lukyanov, O.E.; Chertykovtseva, V.O. *Programma OpenVINT Optimzacii Formy Lopasti Vozdushnogo Vinta [OpenVINT Code for Optimizing the Shape of a Propeller Blade]*; RU 2024610972, FIPS: Moscow, Russia, 2024. Available online: https://new.fips.ru/registers-doc-view/fips_servlet?DB=EVM&DocNumber=2024610972&TypeFile=html (accessed on 20 February 2024).
42. Ali, M.M.; Zhu, W.X. A penalty function-based differential evolution algorithm for constrained global optimization. *Comput. Optim. Appl.* **2013**, *54*, 707–739. [CrossRef]
43. Van To, T.; Ngoc Phien, H. Development of Bezier-based curves. *Comput. Ind.* **1992**, *20*, 109–115. [CrossRef]
44. Zherejov, V.V.; Kusumov, A.N. *Aehrodinamicheskij Raschet Nesushchego Vinta Vertoleta. Uch Posobie Po Kursovomu I Diplomnomu Proektirovaniyu*; KGTU: Kazan, Russia, 1997; 28p. (In Russian)
45. Kravec, A.S. *Kharakteristiki Vozdushnykh Vintov*; Uchebnoe Posobie, Gos. Izd. Oboronnoj Promyshlennosti: Moscow, Russia, 1941. (In Russian)
46. Derksen, R.W.; Rogalsky, T. Bezier-PARSEC: An Optimized Aerofoil Parameterization for Design. *Adv. Eng. Softw.* **2010**, *41*, 923–930. [CrossRef]
47. Espinosa Barcenas, O.U.; Quijada Pioquinto, J.G.; Kurkina, E.; Lukyanov, O. Surrogate aerodynamic wing modeling based on a multilayer perceptron. *Aerospace* **2023**, *10*, 149. [CrossRef]
48. Lin, J.; Yao, H.D.; Wang, C.; Su, Y.; Yang, C. Hydrodynamic performance of a rim-driven thruster improved with gap geometry adjustment. *Eng. Appl. Comput. Fluid. Mech.* **2023**, *17*, 2183902. [CrossRef]
49. Eltaeib, T.; Mahmood, A. Differential Evolution: A Survey and Analysis. *App. Sci.* **2018**, *8*, 1954. [CrossRef]
50. Tanabe, R.; Fukunaga, A.S. Improving the search performance of SHADE using linear population size reduction. In Proceedings of the IEEE Congress on Evolutionary Computation (CEC), Beijing, China, 6–11 July 2014; pp. 1200–1207. [CrossRef]
51. Espinosa Barcenas, O.U.; Quijada Pioquinto, J.G.; Kurkina, E.; Lukyanov, O. Multidisciplinary analysis and optimization method for conceptually designing of electric flying-wing unmanned aerial vehicles. *Drones* **2022**, *6*, 307. [CrossRef]
52. Wong, I.; Liu, W.; Ho, C.H.; Ding, X. Continuous adaptive population reduction (CAPR) for differential evolution optimization. *SLAS Technol.* **2017**, *22*, 289–305. [CrossRef] [PubMed]
53. Iman, R.L. *Encyclopedia of Quantitative Risk Analysis and Assessment*; John Wiley Sons: Hoboken, NJ, USA, 2008; pp. 408–411.
54. Zielinksi, K.; Weitkemper, P.; Laur, R.; Kammeyer, K.D. Examination of Stopping Criteria for Differential Evolution Based on a Power Allocation Problem. Available online: https://api.semanticscholar.org/CorpusID:13358798 (accessed on 2 November 2023).
55. Sharma, H. Lightweight Pipelining in Python. Using Joblib for Storing the Machine Learning Pipeline to a File. Available online: https://towardsdatascience.com/lightweight-pipelining-in-python-1c7a874794f4 (accessed on 2 November 2023).
56. Piotrowski, A.P. Review of differential evolution population size. *Swarm Evol. Comput.* **2017**, *32*, 1–24. [CrossRef]
57. Lukyanov, O.E.; Espinosa Barcenas, O.U.; Zolotov, D.V. Experimental model of an electric power plant for small UAV's automatic control systems. In Proceedings of the 2021 International Scientific and Technical Engine Conference, Samara, Russia, 23–25 June 2021.